Graz University of Technology

Institute for Computer Graphics and Vision

Dissertation

On the Usage of Context for Augmented Reality Visualization

Erick Mendez

Supervisor

Prof. Dr. Dieter Schmalstieg

Graz University of Technology

Advisor

Prof. Dr. Steven K. Feiner

Columbia University

Graz, Austria, November 2010

Abstract

Augmented reality is a user interface paradigm, which mixes virtual and physical information in a single output. Augmenting virtual information on top of physical information at interactive framerates causes a number of issues. These include poor perception of distances, image pollution and attention direction.

In order to address these problems, this dissertation explores the usage of information that is not the focus of the current task but provides a situational reference. This information is often referred to as contextual information or *context*.

This dissertation addresses multiple uses of context for augmented environments. It attempts to encourage the usage of context in augmented reality. It does so by presenting a series of techniques that address a variety of problems, ranging from scenegraph styling to human visual attention. All of the presented techniques are successful in attaining their goal.

The work presented in this dissertation allows the styling of scenegraphs on the fly based on tagged context. It also improves the perception of depth in augmented environments for single occlusion setups. Finally, by carefully manipulating context it is capable of directing the human visual attention to desired portions in the image.

Contents

Chapter 1

Introduction

Augmented reality (Sutherland 1968; Caudell and Mizell 1992; Milgram and Kishino 1994; Azuma 1997) is a user interface paradigm that strives at seamlessly merging digital and physical information by co-locating digital data with the "real world". This enforces a classification of the data: some of it will be of higher importance for the task at hand while the rest is largely ignored. This classification is usually referred to as focus and context. *Focus* information refers to the most important data in the current situation, while *context* refers to extra information that creates a framework or a frame of reference for the focus. Thus, the context portion is typically overlooked and seldom exploited.

The context portion of an augmented reality application typically represents a large amount of information with different degrees of interpretation. One may see spatially close data, or tagged information inside a complex data structure as the context of the scene. This dissertation concentrates on the idea that the context portion of the scene should not be disregarded, but rather exploited, be it by stylizing scenegraphs on the fly, enhancing the perception of depth, or by directing people's attention.

1.1 Augmented reality

Augmented reality, or AR for short, is a human computer interface paradigm within the general computer graphics discipline. AR aims at moving digital information into the physical world, thereby blurring the border between the physical and the virtual in a way that appears natural to the user. It enables a more intuitive, yet complex interface between man and machine. There is, however, no one widely accepted definition of what AR really is.

AR was started by Sutherland with his seminal work on head mounted displays (Sutherland 1968). That work presented the first AR system (see Figure 1). But it was not until the 1990s that an attempt at clearly defining AR was given. Caudell and Mizell coined the term "augmented reality" in 1992 (Caudell and Mizell 1992) and two years later, a first attempt at defining AR was made.

There exist two widely known definitions of AR today. The first, by Milgram and Kishino (see Figure 2), defines AR within the "Reality–Virtuality" continuum (Milgram and Kishino 1994). The second, by Azuma, gives more detailed criteria on what the prerequisites for AR are (Azuma 1997). Azuma's original definition requires:

- A combination of physical and virtual data

- Registration in the physical world in 3D
- Computed interactively/on-the-fly

Combining physical and virtual information is what most people understand for AR. Adding the constraint of three-dimensional registration intentionally rules out all applications that merely display information over a video feed with a disregard of the underlying data, such as the news ticker on news shows. Requesting that the system should be computed on-the-fly differentiates AR from offline computer augmented movies.

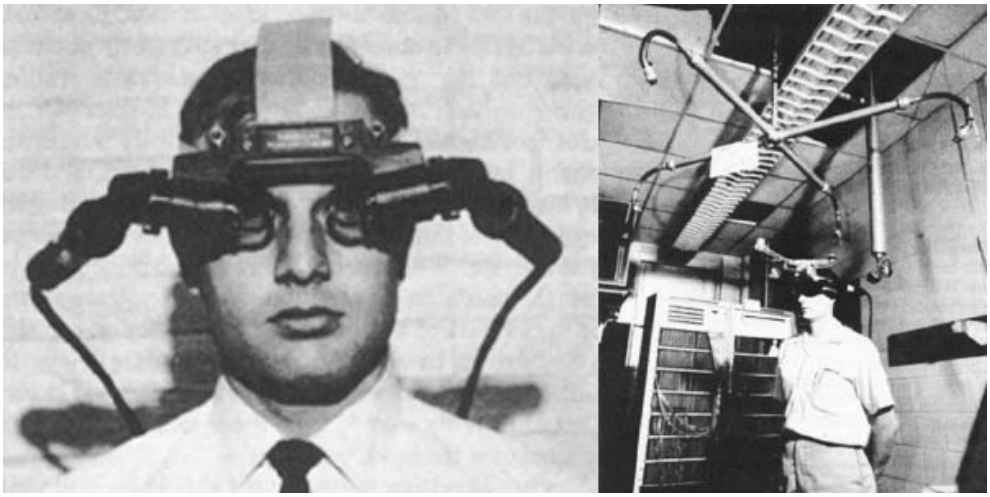


Figure 1. The first see-through head mounted display. Ivan Sutherland's work is agreed to be the first AR system (Sutherland 1968).

Figure 2. Milgram-Kishino continuum. The Milgram-Kishino continuum provides a reference to the domain of mixed reality.

Milgram and Kishino define AR as one possible manifestation within mixed reality that brings physical and virtual information in one display (Milgram and Kishino 1994). Unlike *augmented virtuality*, AR assumes a larger part of the physical (real) information populating the display with a limited portion augmented with virtual data. Augmented virtuality instead, assumes a larger portion to be virtual information augmented with a limited portion of physical (real) data. As it can be seen, this is a loose definition and subject to interpretation.

Mediated reality is another term used in this dissertation, more specifically "non-augmenting" mediated reality (i.e., a technique that does not add artificial virtual objects to the final image). It is generally used to extract or remove information is from the physical data rather than adding to it. Examples of mediated reality are typically focused on the removal of existing physical information such as entire objects or abstract information (e.g., contrast). Early examples addressed the restoration of old photographs or film (Nitzberg, Mumford, and Shiota 1993; Kokaram et al. 1995; Hirani and Totsuka 1996), although the most widely known works are the image in-painting of Bertalmio et al. (Bertalmio et al. 2000) and the augmented interiors by Siltanen and Woodward (Siltanen and Woodward 2006).

This dissertation is based on AR as defined by both Azuma et al. and Milgram and Kishino. Additionally, Chapter 5 describes a technique for attention direction that falls under the definition of mediated reality.

1.2 Context

There exist multiple definitions of what context is. Some definitions give specific examples of context, while others use synonyms. For example, one may define context as the objects in our surroundings (definition by example) or may say that context is the current situation (definition by synonym). Although most people tacitly understand what context is, they find it hard to put into words. The most compelling definition we have found was given by Dey (Dey 2001). Dey's work elucidates an operational definition of context free of binding examples and in addition it discusses ways in which context can be used by context-aware applications.

In his dissertation work, Dey expresses his concern about how poorly computing environments use context (Dey 2000). This has resulted in an impoverished understanding of what context is and how it may be used. He states that context is understood among humans due to the richness of the language we share and our implicit understanding of situational information. These tools humans possess increase the conversational "bandwidth" during communication. Sadly, human–computer communication does not possess these tools and consequently computers are not capable of taking full advantage of the human-computer dialog.

Dey provides the following definition of context:

> *Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.* (Dey 2000).

This clear definition enables a system to freely classify information as whether it is context or not. Notice that the one restriction of the definition is that context must provide a characteriza-

tion of the object's situation. Notice also that Dey uses the term "entity" but in this dissertation we will use the term "focus."

Throughout this dissertation we use this flexible definition to provide different interpretations of what constitutes context. For example, Chapter 4 defines context as importance information from occluding objects (situation).

Dey goes further in his work and provides a definition of context-aware applications:

> *A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task.*

Additionally, context-aware applications can be further classified into distinct categories. Throughout this dissertation, we will be developing context-aware applications for *presentation*.

The sources and usages of contextual information are varied. In this dissertation, we interpret contextual information as coming from tagged textual attributes or visual information coming from a video feed. There exist, however, several more possible sources. Table 1 outlines a few sources and matching usage for AR applications. The columns of the table represent possible sources of contextual information, while the rows list some example uses. Notice that there are more sources–uses combinations than those presented in this dissertation. Relevant work by others is also shown, although some do not refer to the work as context-sensitive. The contributions of this dissertation are marked in the respective boxes and link to the relevant sections. Empty spaces denote areas of research that are yet to be explored. For example, using tracking uncertainties for information filtering is a promising area of research. In each chapter this dissertation we will outline the sources of context and how it is being used. Additionally, Section 1.4 already presents this information in a condensed form.

	Virtual models	System events (e.g., pose, current task)	Tracking Uncertainties	Visually spatially occluding or surrounding information	Tagged textual attributes
Visual Styling		Section 3.3	(Robertson, MacIntyre, and Walker 2007)	(Lu et al. 2010)	Chapter 3
Depth Perception	Section 4.3			Chapter 4	
Attention Direction	(Biocca et al. 2006)	(Bailey et al. 2009)		Chapter 5	(Schwerdtfeger and Klinker 2008)
Label management	(Bell, Feiner, and Hoellerer 2001)	(Bell, Feiner, and Hoellerer 2001)	(Coelho, MacIntyre, and Julier 2004)	(Leykin and Tuceryan 2004) (Bell, Feiner, and Hoellerer 2001)	
Information filtering	Section 3.2.4.5	(Julier et al. 2002)	(Coelho, MacIntyre, and Julier 2004)	Section 4.4	Section 3.2
Information Revealing	Section 4.3	Section 3.2.4	(Coelho 2005)	Section 4.4	Section 3.2.4.1

Table 1. Sources and Usage of Context. This table provides a reference frame for the work presented in this thesis with some example sources of context as well as potential uses. Rows list possible uses, while columns list possible sources. Additionally, related work is also listed. Empty cells denote possible future research directionsa.

1.3 Problem statement

As noted by Dey, the problem is that a large portion of the available information is disregarded. Large geospatial databases, for example, are populated with the results of hundreds of person-years of surveying effort. Utility workers access these databases during fieldwork to help them determine asset location. The stored data has many levels of information. A buried water pipe is not only a collection of geo-referenced locations but it contains information such as owner-

ship, pressure, or last maintenance date. Similarly, buildings are not simply a collection of geometries that detail their shape, but they also include information such as purpose of the building, number of inhabitants, materials used during construction and so on. It is not uncommon for geospatial data that each asset is tagged with dozens of textual attributes with all information that might be needed on site. Typical AR applications concentrate purely on the geometrical information or any other type of data that might yield a visual result, while ignoring all other tagged information.

The problem is not only that contextual data is underused, but it can be even damaged. For example, traditional AR works under the assumption that we can plainly overlay virtual information on top of a physical image. This assumption, however, can cause a number of cognitive problems, such as a poor perception of distances. The central problem is that virtual objects are free from physical laws. Thus, they can be positioned and oriented at any location in the image. Even if we had a perfectly solid constantly tracked position of all virtual information, our brains would still be deceived by conflicting depth cues.

One of the most important monocular depth cues is occlusion. Objects that are farther from the position of the observer can never occlude objects that are closer. However, the goal of AR applications is often that of revealing occluded objects. This causes a conflict: How can we appropriately convey the depth of an object if we remove the occlusion cue (the situational information, or context)?

Overlaid contextual information is not the only disregarded information. Surrounding spatial information can also be exploited for mixed reality applications. Information surrounding the object of interest, not occluding it, provides a frame of reference through which we experience data. Surrounding data that offers a significant contrast to the focus emphasizes its importance. One of the research directions that we explore is that of modifying the information surrounding an object of interest in order to direct our visual attention towards it.

The problem addressed in this dissertation is that of the misuse, underuse, and damage of contextual information. This dissertation explores multiple situations in which using contextual information can provide a richer AR experience.

1.4 Contributions

The contribution of this work is a series of solutions to common AR problems based on usage of contextual information. The solutions presented primarily concentrate on visual and perceptual enhancements for AR. The examples given are explained in detail and offer a clear path for re-implementation. This enables further work to be built on top of context-based solutions.

The core idea remains that of encouraging the use of other sources of information besides the designed focus portion. We do this by providing multiple examples and solutions to common problems in AR applications. The individual solutions target different visual obstacles endemic to the mixing of virtual and physical information.

The end result is a series of techniques that improve the visual and perceptual results of AR applications based on the usage of contextual information. The work presented in this dissertation is built on the Studierstube project (see Section 2.1.1). Studierstube provides a basic framework for the fast prototyping of AR applications. The solutions throughout this dissertation are implemented as part of the Studierstube components toolset, thereby extending the framework.

1.4.1 Contributions to scenegraph styling

Chapter 3 is centered on the idea that even data that does not provide a direct visual result can be exploited, for example, for information filtering or information revealing. The source of contextual information in that chapter is tagged textual attributes transcoded from geospatial databases. The core usage is the stylization of subgraphs upon runtime. The basic idea is that high density information leads to display clutter, thus, information filtering techniques are investigated as a remedy.

In practice, AR enables the possibility of changing the object of interest at runtime. For example, a tired tourist's interest may change from cultural landmarks to public transportation and the route back to the hotel. This dynamic changing of the interest in objects increases the complexity of the application implementation and design. Traditionally, application design demands that designers have to decide on the range of possible objects of interest and their respective styles beforehand. If a scenegraph-based implementation is used, the application code must be tightly coupled with the scenegraph data structure so that procedural calls can modify the rendering styles of the right portions of a scene graph.

What Chapter 3 proposes is a stylization of portions of the scenegraph at runtime based on contextual information. This allows the scenegraph to be more than just pure geometrical information, but instead be enriched with textual attributes. The concept is similar to the work of Beach and Stone on graphical style sheets (Beach and Stone 1983) which allowed basic styling of geometries. The usage examples presented in Chapter 3 are quite varied. For example, Section 3.2 largely focuses on information filtering through the usage of magic lenses (Bier et al. 1993) combined with context-sensitive scenegraphs (Reitmayr and Schmalstieg 2005). Figure 3 shows an example of the visual effects possible with context-sensitive magic lenses (introduced in Section 3.2.2). The figure shows a foam model of downtown Graz tracked by ArtoolkitPlus markers. The image on the right shows the model overlaid with gas pipes and electrici-

ty lines extracted from a real world database. The result is a cluttered view in which it is impossible to distinguish individual assets. The image on the right shows the same model using a magic lens to only show electricity lines. As it can be seen the image is less visually polluted.

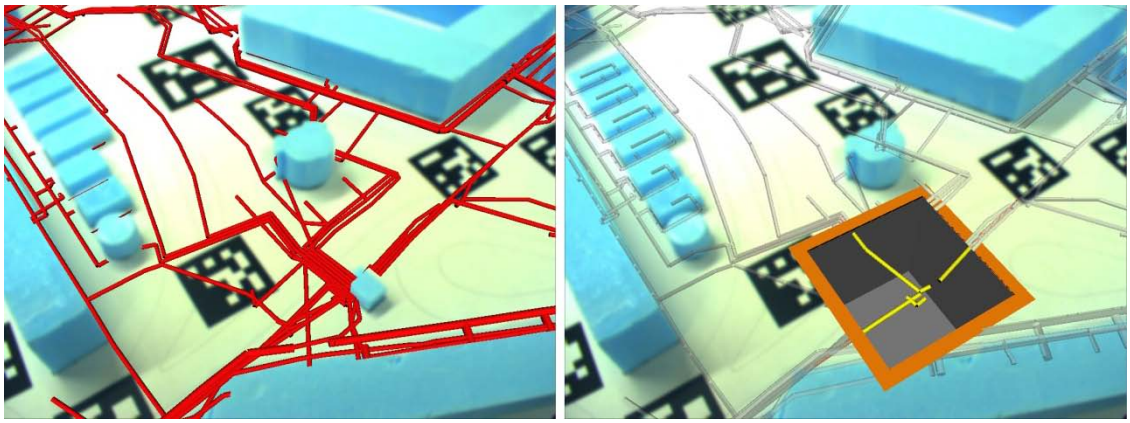


Figure 3. Example usage of filtering by contextual information. Chapter 3 will present in detail ways in which tagged contextual information can be used to reduce visual clutter.

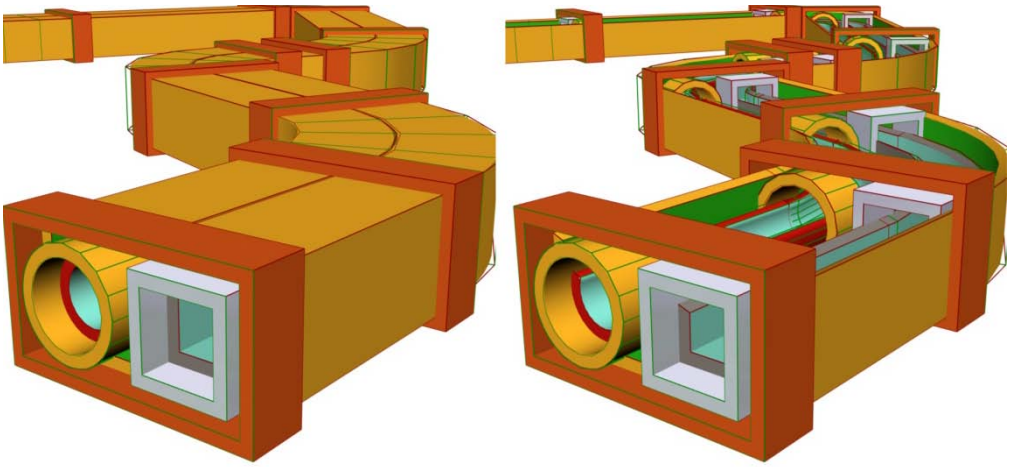


Figure 4. Example procedural generation of models based on contextual information. Chapter 3 will introduce the idea that geometrical models can be generated based on contextual information extracted from geospatial databases.

Section 3.4 uses contextual information to influence the procedural generation of models at runtime. It is a combination of an engine for the procedural generation of models (Havemann 2005) and the context-sensitive scenegraph (Reitmayr and Schmalstieg 2005). Figure 4 shows an example of the visual effects possible if the generation of geometrical models is influenced by contextual information. The figure shows two versions of the same scenegraph. Both images contain two pipes (one with a circular cross-section, one with a square profile) encased by a third object with a rectangular profile. The image on the left shows the objects with a completely closed profile while the image on the right shows the same assets with an open profile.

Whether the profile of the objects in the scene is closed or open is bound to a contextual attribute, in this case coming from a geospatial database. The engine for the procedural generation of models is itself a node and can thus be influenced by the proposed context-sensitive scenegraph.

The main contributions of Chapter 3 are multiple uses of contextual information for the stylization of scenegraph nodes. These uses span information filtering, information revealing, runtime binding of styles and the procedural generation of models.

1.4.2 Contributions to depth perception

Chapter 4 is centered on the classical problem of depth perception in mixed reality. It puts forward the idea that occluding information should be carefully managed and not merely overlaid, as is typically the case. The source of contextual information is the occluding object itself, whether from video or a registered three-dimensional model. The usage is the preservation of this context to improve the perception of spatial arrangements and of depth for hidden objects. The basic idea is that occluding objects present valuable information to the perception of the scene and should thus be treated with care.

AR displays provide extra information to a user's perception by overriding parts of the physical world with synthetic and computer generated images. However, the heedless replacement of portions of the real world image can easily cause a number of cognitive problems. We consider that if too much information is added in areas that are unimportant in the original image, the impression of a cluttered display is caused. Chapter 4 considers problematic augmentations in X-Ray visualizations, where hidden structures are rendered on top of visible objects. Careless augmentation with synthetic imagery may obscure important information of the physical world. We consider the case where this important information enables the user to infer spatial relationships among objects.

This problem is illustrated in Figure 5. The image on the left shows the result of merely overlaying a virtual object on top of a video feed. Although the pose returned by the tracking correctly indicates the intended position of the virtual model is inside the box, the visual result seems to have the virtual object (the car) painted outside rather than inside. The image on the right shows one of the solutions presented in Chapter 4. In this case a predefined mask is used to indicate the portions of the physical scene that can be overlaid and the amount in which they can be overlaid. The visual result indicates a better containment of the virtual object inside the physical box. The solutions in that chapter are quite varied, some requiring virtual models, others preprocessing steps, others complex engines and so on.

What Chapter 4 proposes is to use contextual information to improve the perception of depth in AR. Chapter 3 already introduced the notion of information revealing by using a magic lens.

The difference between the two solutions is illustrated in Figure 6. Notice that the image on the left presents a magic lens to show the insides of the office, but the large size of the lens is diminishes its effectiveness. In contrast, the image on the right shows a better solution that depends on an analysis of the video feed.

Figure 5. Example usage of context preservation to enhance the perception of depth. Chapter 4 will introduce in detail multiple ways in which contextual information can be used to enhance the perception of depth. (Left) This image shows the problem of careless overlay of virtual objects in an AR scene. (Right) Shows one of the solutions proposed in Section 4.5.1.

Figure 6. Comparing techniques from Chapter 3 and Chapter 4. (Left) X-ray view by using a magic lens. (Right) X-ray view by context-preservation.

Section 4.3 provides the first solution based on an analysis of a linked virtual three-dimensional model. The core idea is to assign a virtual model to all occluding physical objects. Then extract the most important portions of that model and overlay them on the screen in an abstracted manner.

Section 4.4 provides a solution based on an online analysis of the video feed. The core idea is to extract important features from the video feed, then use them as a mask. This mask indicates to the system which parts of the video feed should not be overlaid by augmentations. Unlike

the technique from Section 4.3, this does not require a previous knowledge of the occluding objects in the scene.

Section 4.5 provides two solutions, one based on a predefined importance mask, and the other based on a procedural generation of the mask. Section 4.5.1 presents the first solution, the predefined mask. It allows the most visually compelling results throughout the whole chapter, and it is the most computationally efficient. However, it does require knowledge of the potentially occluding objects. Section 4.5.2 presents the second solution, the procedurally generated mask. It allows for animated masks to constantly change depending on time, or the position of the viewer. The visual results of this engine will depend entirely on the implemented procedural engine. In our case we implemented an engine based on a turbulence function (Perlin 1985).

The main contributions of Chapter 4 are four techniques to improve the perception of depth on augmented environments. All of the techniques used visually occluding information as the context information for the scene. Chapter 4 compares all of the techniques in possible usages, advantages and disadvantages.

1.4.3 Contributions to attention direction

Chapter 5 is centered on the problem of unobtrusively directing human attention in mixed realities. It puts forward the idea of manipulating the contrast of surrounding information in order to promote the salience of certain objects in the scene. The source of contextual information is data that spatially surrounds the focus object. The first usage is to direct human visual attention. The second is to achieve this direction without people noticing that modulation has taken place. The basic idea is to manipulate local contrast features in order to reduce the contrast of undesired areas and to increase the contrast of areas to which we want to direct the attention.

In mixed environments, one has the possibility of directing the viewer's attention by overlaying augmenting artifacts, such as arrows or circles. These types of augmentations, although effective, increase the visual pollution of the final image. Due to the control we have on the input video feed, what we try instead is to create an attention direction technique that does not increase visual pollution and can potentially pass unperceived by the viewers.

What Chapter 5 proposes are two non-augmenting techniques for attention direction. The goal is to manipulate salient features of a video feed in order to increase the chances of a particular portion of the scene to call our attention. We modify image properties such as brightness and saturation on a per-pixel basis. One of the constraints of the technique is to automatically modify the image upon runtime without a prior knowledge of its contents. The final goal of the

technique is to capture the user's attention faster and retain it for a longer time, for a significantly larger percent of the population.

Figure 7 shows a screenshot from a video clip before (left) and after modulation (right). This image is obtained from the videos used for the user study performed in Section 5.3.3. As one can see, the modifications to the image are barely perceptible. However, the modulations effectively drew the attention of participants in our study. That chapter illustrates the problems where this modulation might fail and possible research directions.

Figure 7. Attention direction example. (Top Left) Original capture image. (Top Right) Image after our modulation technique. The goal is to direct the attention to the far right trashcan and water pipe. (Bottom) Enlargements of the focus of interest.

Section 5.2 provides the first solution for attention direction. It is based on modulation of the image in HSV color space. The core idea is to manipulate the dimensions of lightness, saturation and color-opponents in order to reduce contrast of undesired areas. The technique is tested in two ways, first on a numerical form in which we detect the percentage of pixel difference before and after modulation. The second test is a user study with 30 participants. This study is on static images taken with a simple camera. The results are that the technique can effectively draw the attention of the participants albeit damaging the image.

Section 5.3 provides the second solution for attention direction. It is based on modulation of the image in the CIEL*a*b* color space. The core idea is to manipulate brightness and color-opponents in order to simultaneously reduce the salience of undesired areas and increase the salience of desired ones. The validation step is a bit more complex than that of Section 5.2. It takes as input video clips instead of static images. The first goal of this technique is to find the highest strength of the modulation that can be applied without the participants noticing that modulation has taken place. The second goal is to verify whether the modulation effectively draws the attention of participants to desired regions. The results are that the technique can

effectively draw the attention of the participants without them noticing that modulation has taken place.

The main contributions of Chapter 5 are two techniques for attention direction on mixed realities. The techniques are tested with an overall total of 96 participants. Both techniques proved successful and open various avenues for further research on human visual attention direction.

1.5 Collaboration statement

The work presented in here could not have been possible without the help of multiple researchers throughout many publications. The following people have had an impact on the direction of the work presented here and deserve mentioning:

- **Denis Kalkofen** of Graz University of Technology co-developed and co-designed the work presented on Sections 3.2, 4.3, and 4.4.
- **Eduardo Veas** of Graz University of Technology collaborated on the design and performance of the second user study for attention direction from Section 5.3.3.
- **Gerhard Schall** of Graz University of Technology collaborated on the transcoding step described in Section 3.4.1.1.
- **Daniel Wagner** of Graz University of Technology provided guidance and support for the entire marker-based tracking examples.
- **Sven Havemann** of Graz University of Technology collaborated on the integration of the generative modeling language described in Section 3.4.

Although some portions of this dissertation have not been published, the larger part has been composed from a number of peer-reviewed publications.

The work presented in Section 3.2 is based on the following publication:

E. Mendez, D. Kalkofen, and D. Schmalstieg, "Interactive context-driven visualization tools for augmented reality," *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, Santa Barbara, USA: IEEE Computer Society, 2006, pp. 209-218. Reference: (Mendez, Kalkofen, and Schmalstieg 2006).

The work presented in Section 3.3 is based on the following publication:

E. Mendez and D. Schmalstieg, "Context Sensitive Stylesheets for Scenegraphs," *International Journal of Virtual Reality*, vol. 7, 2008, pp. 77-84. Reference: (Mendez and Schmalstieg 2008).

The work presented in Section 3.4 is based on the following publication:

E. Mendez, G. Schall, S. Havemann, S. Junghanns, D.W. Fellner, and D. Schmalstieg, "Generating Semantic 3D Models of Underground Infrastructure," *IEEE Computer Graphics and Applications*, vol. 28, 2008, pp. 48-57. Reference: (Mendez et al. 2008).

The work presented in Section 4.3 is based on the following publication:

D. Kalkofen, E. Mendez, and D. Schmalstieg, "Interactive Focus and Context Visualization for Augmented Reality," *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, Nara, Japan: IEEE Computer Society, 2007, pp. 191-200. Reference: (Kalkofen, Mendez, and Schmalstieg 2007).

The work presented in Section 4.4 was based on the following publication:

D. Kalkofen, E. Mendez, and D. Schmalstieg, "Comprehensible Visualization for Augmented Reality," *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, vol. 15, 2009, pp. 193-204. Reference: (Kalkofen, Mendez, and Schmalstieg 2009).

The work presented in Section 4.5.1 is based on the following publication:

E. Mendez and D. Schmalstieg, "Importance masks for revealing occluded objects in augmented reality," *ACM Symposium on Virtual Reality Software and Technology (VRST)*, Kyoto, Japan: 2009, pp. 247-248. Reference: (Mendez and Schmalstieg 2009).

The work presented in Section 5.2 is based on the following publication:

E. Mendez, D. Schmalstieg, and S.K. Feiner, "Experiences on Attention Direction through Manipulation of Salient Features," *IEEE Virtual Reality (VR) Workshop on Perceptual Illusions in Virtual Environments*, Waltham, MA USA: 2010, pp. 4-9. Reference: (Mendez, Schmalstieg, and Feiner 2010).

The work presented in Section 5.3 is based on the following publication:

E. Mendez, S.K. Feiner, and D. Schmalstieg, "Focus and Context in Mixed Reality by Modulating First Order Salient Features," *International Symposium on Smart Graphics*, Banff, Canada: Springer, 2010, pp. 232-243. Reference: (Mendez, Feiner, and Schmalstieg 2010).

1.6 Organization

This dissertation presents several techniques to exploit the existence of contextual information in an AR scene. The resulting work varies as to the definition of what contextual information is and for what purpose it is used. Although multiple techniques will be presented throughout this dissertation, they have been classified by their definition of what is context and by the final goal of the technique.

The dissertation first provides a set of background information that the reader should be familiar with and then progressively presents the developed techniques. Finally, it concludes with an overview of the main contributions of the work and the lessons the author has learned throughout the development of this work. The dissertation is then structured as follows:

- **Chapter 2 Related Work** presents a collection of scientific relevant publications. It provides a survey of other work that has attempted a similar task to that presented here, but with different strategies. Moreover, it provides background information that the reader should understand to better appreciate the subsequent contributions.

- **Chapter 3 Using Context for Scenegraph Styling** introduces techniques that exploit existing contextual information for styling scenegraphs on the scene during runtime. It uses the concept of context as tagged textual data that resides in the data structure itself.

- **Chapter 4 Using Context to Enhance the Perception of Depth** presents the concept of improving the perception of depth in mixed reality environments by manipulating the contextual portion of the scene. This chapter uses the concept of context as data that is spatially occluding the focus information.

- **Chapter 5 Using Context for Visual Attention Direction** introduces the idea that one may manipulate the human visual attention by carefully modifying the contextual portion of the scene. This chapter uses the concept of context as data that is spatially close to the focus information.

- **Chapter 6 Conclusion** finally offers remarks, lessons learned and possible future research directions for the multiple techniques presented in this dissertation.

Chapter 2

Related Work

The work presented in this dissertation revolves around the general paradigm of AR within computer graphics. It addresses multiple topics such as human computer interfaces, visual perception and scenegraph management. We will now present the previous work most relevant to this dissertation; however, more detailed technical aspects will be examined in subsequent chapters.

Due to the diversity of the research presented in this dissertation, the related work is equally diverse. We will present the most relevant related work in subsections relating to the remaining chapters. Section 2.1 will introduce general concepts and related work that the reader should be familiar with for the whole dissertation. Section 2.2 will introduce concepts and related work that the reader should be familiar with before reading Chapter 3. Section 2.3 will introduce general concepts and related work that the reader should be familiar before reading Chapter 4. And finally, Section 2.4 will introduce general concepts and related work that the reader should be familiar with before reading Chapter 5.

2.1 General related work

The following is a list of concepts and related work that the reader should be familiar with in order to understand the work presented in this dissertation.

2.1.1 Studierstube

The basic software platform used in this dissertation is the Studierstube AR project (Schmalstieg et al. 2002). Studierstube provides the design foundation of all the contributions presented in Chapter 3, Chapter 4 and Chapter 5. For that reason it is important to describe its main concepts and features.

Studierstube has been under development since 1996. Its goal is to provide a groundwork upon which AR applications can be easily built. In essence, it is a set of nodes extending the Open Inventor rendering library (Strauss and Carey 1992) plus a set of classes that provide runtime support. It includes support for three-dimensional interaction, propagation of tracking events, access to the lower level OpenGL API, as well as all necessary tools for rendering of mixed reality applications.

Two related projects provide additional assistance to basic AR functions. Opentracker is an open software architecture that provides a generic solution to the different tasks involved in tracking input devices and processing tracking data for virtual environments. It combines a

highly modular design with a configuration syntax based on XML (Reitmayr and Schmalstieg 2001). Openvideo is an open software architecture that provides a generic solution to the different tasks involved in the capture and presentation of video feeds. Similar to Opentracker, it combines a highly modular design with a configuration syntax based on XML (Kalkofen et al. 2006).

Studierstube is a component-oriented software architecture that addresses the specific needs of AR. Its goal is to provide reusable high performance components for AR that are sufficiently general to be used across different AR applications. All of the examples in this thesis are implemented as Studierstube components that may be re-used by other AR applications. For example, Chapter 3 provides a component for stylizing a scenegraph depending on propagated data. Chapter 4 highlights portions of the compositing framework, a component for the scripting of advanced rendering techniques. The same compositing framework is then used by Chapter 5 to setup a rendering pipeline based on fragment shaders. A shader is a set of software instructions used primarily to calculate rendering effects on graphics hardware with a high degree of flexibility. Shaders are used to program the graphics processing unit (GPU) programmable rendering pipeline. A fragment is the data necessary to generate a single pixel's worth of a drawing primitive in the frame buffer.

2.1.2 Computer graphics concepts

Although the particular topic addressed in this dissertation is AR, the main computer science discipline is computer graphics. Thus, the reader must be familiar with concepts such as rendering, modeling, color spaces and so on. This dissertation will make use of terms largely explained by Foley et al. (Foley et al. 1995). Additionally, it is recommended that the reader be familiar with OpenGL programming (Shreiner et al. 2005).

Chapter 4 will make extensive use of OpenGL shading programs (Rost 2004). In particular vertex and fragment shaders (Rost 2004) – geometry shaders are not used in this dissertation. It is recommended that the reader be familiar with the OpenGL Shading Language (GLSL) (Rost 2004). It is also important for the reader to have a basic understanding of the OpenGL rendering pipeline (Shreiner et al. 2005).

2.1.2.1 Scenegraph

Scenegraphs are widely used throughout this dissertation. A scenegraph is a data structure often used by rendering engines (Wernecke 1994). It is typically an acyclic graph. Application designers arrange the elements in a graphical scene inside the scenegraph. Since the scenegraph is a tree graph an operation applied to a parent node is propagated to all its children nodes. For example, applying a transformation to a node will consequently be propagated to all its children that result in a typically efficient and natural way to process these

operations. Thus, a common feature is to logically group shapes inside a node that can later be rotated, scale or moved as if they are a single object.

Traversal is a technique in which all the nodes of a graph are visited and operations are applied to them (Drozdek 2004). It is a powerful tool used for scenegraphs. A traversal typically starts at the root node of the scenegraph and recursively visiting the children of each node while performing an operation (such as a transformation) until all nodes have been visited.

2.2 Related work for scenegraph styling

The following is a list of concepts and related work that the reader should be familiar with in order to understand the work presented in Chapter 3.

2.2.1 Focus and context

Focus and context (F+C) visualization is a family of computer graphics techniques for focusing a user's attention. They allow applications to direct the attention of the user to a portion of the data (focus), while at the same time the overall situational relationship of the neighboring information (context) is presented in an abstracted way. Following the overview of focus and context techniques given by Kosara et al. (Kosara, Hauser, and Gresh 2003), the creation of focus and context visualizations can be roughly divided into two major steps: data classification (that includes interaction with the user) and rendering. The objective of the first step, data classification, is to identify the focus and context roles in the data (i.e., what information should be focus and what should be context). Data classification may be statically given for a particular application, or it may be determined procedurally by some kind of interpretation procedure. Data classification is usually performed based on user input, so the choice of focus can be controlled by the user. There are a number of approaches to how the user can be involved in the definition of context; for example, through direct pointing, widget manipulation or tangible interfaces.

Once having distinguished between focus and context, the second step is to render each of the two categories in visually distinctive styles in order to direct the user's attention to the focus. This may be done by modifying geometry (e.g., visually distorting the data through a magnifying lens), or by modifying appearance (e.g., by manipulating saturation or opacity). Most of the techniques for applying focus and context rendering to three-dimensional scenes rely on opacity modification. Several other projects use interesting rendering changes to attract the attention of the user, such as color or depth of field discrimination or modifying shading parameters. We draw inspiration from this work, since our framework allows very general non photorealistic effects to be applied to AR scenes.

2.2.2 Magic lenses

Magic lenses were first introduced as a user interface tool in 1993 by Bier et al (Bier et al. 1993). They are filters that modify the presentation of scene objects in a locally bounded area. Magic lenses can be used to reveal hidden information, to enhance data of interest, or to suppress distracting information. Figure 8 shows examples of the original magic lenses.

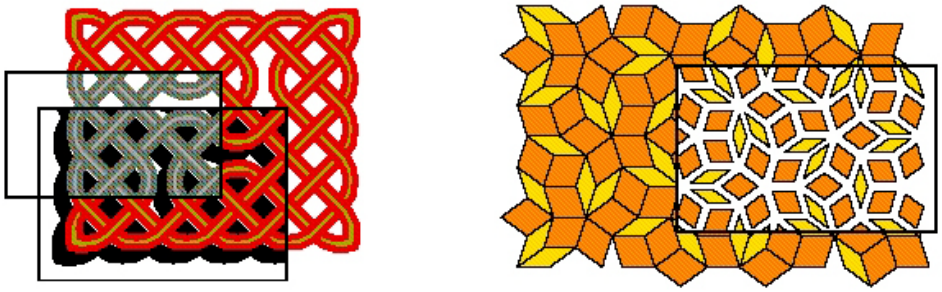


Figure 8. Original magic lenses. These magic lenses were introduced by Bier et al. (Left) Two magic lenses shown, the first turns objects to grayscale, the second adds a shadow. (Right) The magic lens adds space between tiles (Bier et al. 1993).

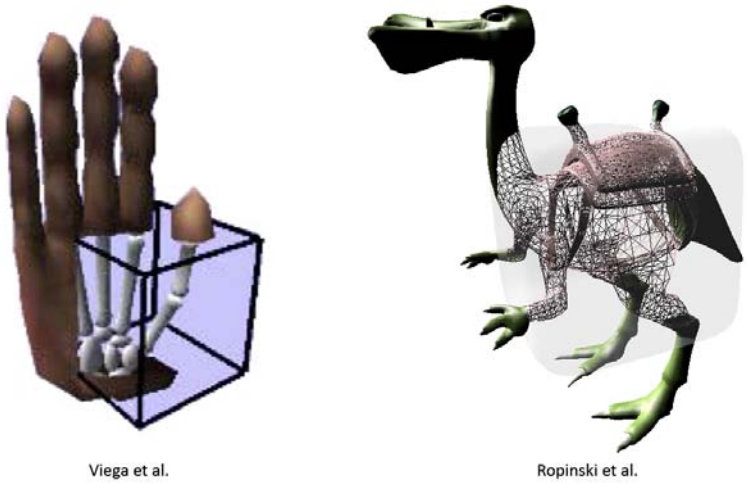


Figure 9. Three-dimensional magic lenses. (Left) An example of the three-dimensional magic lens as defined by Viega et al. (Viega et al. 1996). (Right) An example of the three-dimensional magic lens as defined by Ropinski and Hinrichs (Ropinski and Hinrichs 2004).

These lenses were later extended to three dimensions. Figure 9 shows examples of three-dimensional magic lenses. The image on the left shows the work that Viega et al. developed for flat and volumetric magic lenses for three-dimensional environments (Viega et al. 1996). Viega et al. provide an interesting discussion on the rendering technique of flat lenses, and how multiple lenses can be combined. However, overlapping volumetric lenses are not discussed. Another algorithm for the rendering of three-dimensional magic lenses was later presented by Ropinski and Hinrichs (Ropinski and Hinrichs 2004) – image on the right. That

technique uses multi-pass rendering to achieve the effect of a magic lens. A more detail description of the algorithm by Ropinski and Hinrichs is given in Section 3.2.1.

Magic lenses have been extensively used in computer graphics. For example, in mixed reality, Looser et al. presented an interesting work that mixed the use of lenses and semantic information (Looser, Billinghurst, and Cockburn 2004). The interaction techniques discussed in their paper are magnification, object selection and information filtering. They have also been implemented for volume rendering by Wang et al. (Wang et al. 2005). That tool allowed them to apply a number of free-style lenses, which conveniently emphasized specifics parts of the visualized data without losing the overall context. It must be noted that this work uses context as the overall visual relationship of the displayed data with its surroundings. Thus, for example, arbitrary subsets of data in the same set (such as bones or tissue in an MRI scan) could not be treated differently.

2.3 Related work for depth perception

The following is a list of concepts and related work that the reader should be familiar with in order to understand the work presented in Chapter 4.

2.3.1 Depth perception

The basic definition of Azuma (Azuma 1997) (see Section 1.1) leaves open a variety of problems to be addressed for AR, such as how to achieve interactive frame rates, what accuracy for 3D registration is necessary, and how to properly mix virtual and physical information. This last question is the main topic addressed in Chapter 4.

Furmanski et al. note the problem that it is difficult to perceive the three-dimensional location of an object with a single two-dimensional planar projection (see Figure 10) (Furmanski, Azuma, and Daily 2002). The problem of mixing virtual and physical information goes beyond photorealistic rendering. One might be tempted to assume that if virtual information was rendered with the highest possible fidelity, it would perfectly and effortlessly mix with physical information. Although this would definitely give the user a better experience, the problems faced by AR are not based solely on color or illumination, but largely on the perception of distances. While there is encouraging work on see-through displays, polarized shutter glasses, and three-dimensional displays, much of the commercial and scientific work on AR is performed on monocular displays with physical data captured by monocular cameras. This reduces the problem of mixing physical and virtual data to mixing two two-dimensional images –although, Kalkofen et al. present the concept of mixing information with multiple layers (Kalkofen, Mendez, and Schmalstieg 2007).

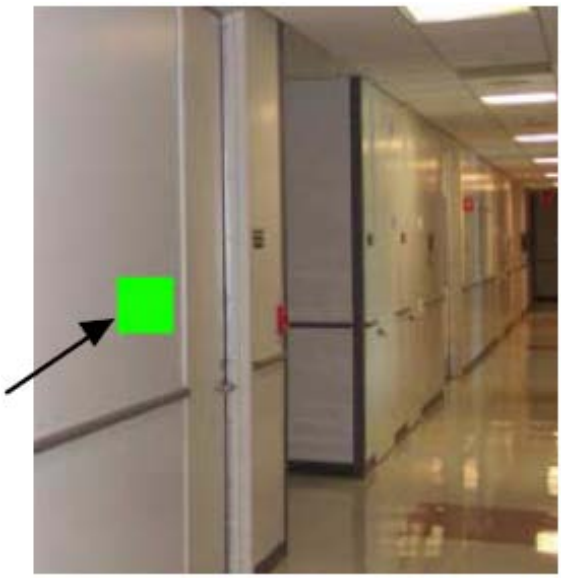


Figure 10. Depth ambiguity problem as defined by Furmanski et al. In this display, the location (in depth) of the rendered square (marked with an arrow) is ambiguous. Is it on the surface in the hallway, to right of the near room? In the near room? In the far room? There is no definite way to tell because 2D-planar projections of three-dimensional space can be ambiguous (Furmanski, Azuma, and Daily 2002).

In order to perceive depth, the human visual system uses a number of cues. These are typically classified into binocular cues that require input from both eyes and monocular cues that require the input from just one eye (Goldstein 2006). Binocular cues exploit the visual disparity between the two eyes' images, as well as the muscular effort to focus on a point by rotating each eye in the opposite direction (vergence). Monocular cues are more numerous and include the relative size of objects, the amount of texture detail, linear perspective, shifts on the color hue (farther objects shift towards blue), highlights, shadows, and monocular motion parallax. Some of these cues may be synthetically generated by rendering the virtual data if enough depth information is present.

The depth information of the virtual two-dimensional image is generated by the graphics pipeline. But the depth information of the physical video image is not present. There exist only few devices capable of delivering both depth information and video feed, although this trend seems to be rapidly changing, for example, with the introduction of the Microsoft Kinect (Microsoft 2010) project. In the work presented in this dissertation we will only consider AR applications that receive the physical information from a monocular video camera. In any case, even if depth information from the physical camera was present, the problem remains on how to mix it with virtual data. In AR applications, it is often the case that virtual information lies behind, inside or at the same place as their physical counterpart. The techniques and solutions that address the problem of mixing two or more two-dimensional im-

ages with only partial depth information are often referred to as *occlusion-management techniques*.

Occlusion management has been a constant topic of research in computer graphics particularly in virtual and augmented reality. Studies have shown that merely overlaying a virtual representation of the object that is about to be occluded disrupts the perception of distance and spatial arrangements across the objects in the scene (Interrante, Fuchs, and Pizer 1996). This means that carelessly overlaying virtual information merely swaps the roles of the occluding and hidden objects.

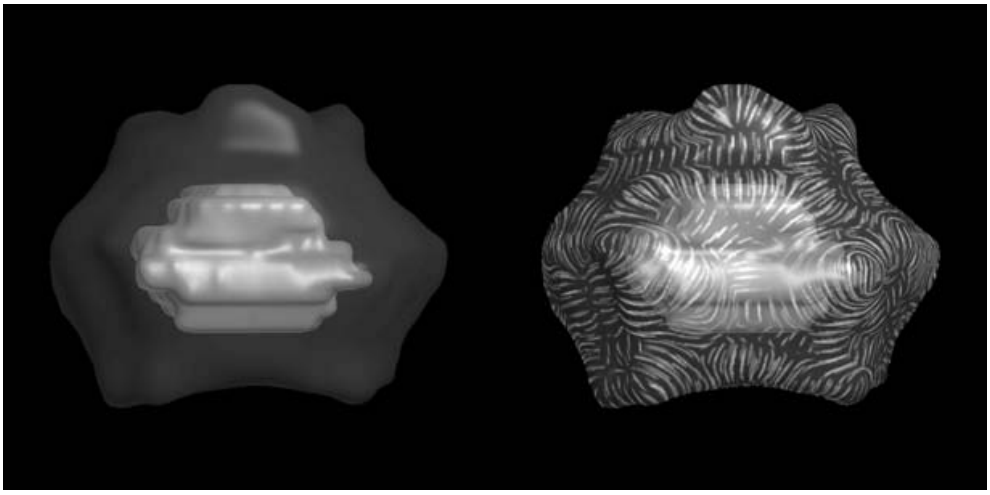


Figure 11. An illustration of curvature directed strokes. (Left) Object encased by a transparent smoothly curved surface. (Right) The transparent object has been enhanced with curvature directed strokes (Interrante, Fuchs, and Pizer 1996).

Interrante et al. have conducted thorough studies on the effects and possible solutions for occlusion management (Interrante, Fuchs, and Pizer 1996; Interrante, Fuchs, and Pizer 1997). They have focused on the problem that it is often difficult to adequately perceive the three-dimensional shape of a layered transparent surface or its relative depth distance from underlying structures. The solution presented suggest that uniformly distributed opaque short strokes, locally oriented in the direction of greatest normal curvature, and of length proportional to the magnitude of the surface curvature in the stroke direction aid the perception of distances (see Figure 11). The main concept is that not all information from the originally occluding object should be disregarded; instead, there exist a number of its attributes that should be preserved. What these attributes are is a question addressed in Chapter 4.

The concept of the preservation of features from the occluding object was brought to volume rendering by Kruger et al. (Kruger, Schneider, and Westermann 2006). The Clearview project enabled users to focus on particular areas of a volume while preserving contextual

information (from occluding regions) without visual clutter. We expanded the notion into the AR domain by mixing physical and polygonal data (Kalkofen, Mendez, and Schmalstieg 2007). At the same time Bichlmeier et al. presented a similar concept by mixing physical and volumetric data (Bichlmeier et al. 2007). The contextual anatomic mimesis overlaid medical CT data on top of deceased and live subjects. They preserved information from the occluding subjects by retaining edges and shadows from the video feed.

These are, however, not the only techniques for ameliorating the errors on depth perception. In the past there have been multiple attempts such as view restrictions. View restrictions, as the name suggests, limit the area on which virtual data may be seen. This restriction typically is aligned with the surface of the occluding object, thus exploiting motion parallax when the user moves. Restrictions are not considered in Chapter 4, but are exploited in Section 3.2. Other techniques include transparency modulation, or cutting parts of the occluding object away such as the work by Feiner and Seligmann (Feiner and Seligmann 1992). Other solutions spatially change the arrangements of virtual objects in the scene, such as with explosion diagrams (Li, Agrawala, and Salesin 2004). None of these techniques, however, are considered throughout this dissertation.

2.3.2 Edge extraction and stylization

Interrante et al. argue that in order to better convey the containment of one object inside another, curvature directed strokes from the occluding object should be preserved (Interrante, Fuchs, and Pizer 1996). Chapter 4 explores this idea further with multiple techniques based on the shape of the occluding object. Most of these techniques are based on the assumption that the most important portions of the occluding object should be preserved. The work of Interrante et al. shows that the portions to preserve are typically those that somehow convey the shape of the occluding object. Although they experimented with curvature directed strokes, many other possibilities exist.

There exists a large amount of work on non-photorealistic (NPR) rendering to suggest shape rather than explicitly represent it. This is commonly done by computer generated line drawings. For example, DeCarlo et al. propose the idea of using suggestive contours in addition to silhouettes and contours for conveying shape. Suggestive contours are distinct from creases and lines along ridges or valleys. They define suggestive contours as an extension of contours to account for "nearby" viewpoints. The silhouettes and contour curves of a shape are two-dimensional projections of its three-dimensional points whenever the normal of the point is orthogonal to the viewpoint (Koenderink 1990). Contour lines act as markers for internal discontinuities in depth in a two-dimensional image. Silhouettes actually separate the foreground object from the background. Ridges and valleys are lines that are indepen-

dent of the viewpoint and correspond to the places on an object where the surface normal changes direction rapidly.

However, these edges are all extracted from geometry rather than a two-dimensional image. Most of the work in this dissertation relies on edge extraction from a two-dimensional image. For example, Section 4.3 extracts edges from a rendered object with a Canny edge detector (Canny 1986) while Section 4.4 uses the image variance.

The Canny edge detector is a multi-pass process. To detect edges, the Canny edge detector first smoothes the image with a Gaussian convolution. It then computes the gradient to find high spatial first derivatives. It subsequently follows along these edges and zeroes the pixels that are not actually on the edge border in order to thin the output. This whole process is quite computationally demanding for applications that require interactive framerates. To reduce the amount of computational power, we also use the image variance. The variance of an image is a measure of its statistical dispersion, indicating how far from the expected value its values typically are. It can be computed in a single rendering pass to extract edges from an image.

Many other edge detection techniques exist in the literature. Heath et al. provide a good comparison of multiple techniques (Heath et al. 1996). In that paper, they compare edge detectors from Canny (Canny 1986), Bergholm (Bergholm 1987), Iverson and Zucker (Iverson and Zucker 1995), Nalwa and Binford (Nalwa and Binford 1986), and Rothwell et al. (Rothwell et al. 1995). Any of these edge detectors could potentially be used for the techniques presented in Chapter 4, however, due to its simple implementation, out of these only the Canny edge detector is used.

2.4 Related work for attention direction

The following is a list of concepts and related work with which the reader should be familiar in order to understand the work presented in Chapter 5.

2.4.1 Visual attention

The final challenge of this dissertation is to influence the visual attention of users when looking at either images or videos. In order to achieve this, one must first understand how human visual attention works. This section provides background information on the attention process, visual salience and techniques to exploit it.

Attention refers to the process by which organisms select a subset of available information upon which to focus for enhanced processing and integration (Ward 2008). It has evolved in complex biological systems to rapidly detect predators, prey or potential reproductive mates. Attention is a solution to our sensory organs being able to provide more information

than our brains can simultaneously process. It is considered to involve at least three aspects: orienting, filtering and searching.

Orienting, for example, happens when we hear a loud noise behind us and we turn our sensing organs (eyes and ears) in the direction of the noise. *Filtering* involves our active discarding of information, such as one does while reading email during a conference talk. *Searching* is used when we must find an item in our vicinity, for example, while looking for difficult to find food items.

2.4.2 Visual salience and the saliency map

In an image, an object is said to be visually salient if it stands out more than its surrounding neighborhood (Lee, Kim, and Choi 2007). The saliency at a given location is determined primarily by how contrasting this location is from its surrounding in dimensions such as color, orientation, motion, and depth (Koch and Ullman 1985). The conspicuities of a location are measures that represent how contrasting this location is to its surroundings in each of said dimensions (Itti, Koch, and Niebur 1998). Treisman and Gelade use the term 'dimension' to refer to the range of variations, and 'feature' to refer to values in a dimension, for example, color and lightness are dimensions, while yellow and dark are features (Treisman and Gelade 1980). Thus, a black object on a white background has a high lightness conspicuity, but a low color conspicuity. The visual salience of a location is the combination of all its conspicuities. A scene's saliency map is a map of the salient values on each location in the image. Thus, v*isual saliency* (or *visual salience*) is the distinct subjective perceptual quality that makes some items in the world stand out from their neighbors and immediately grab our attention (Itti 2007). It refers to the evolved process on primates and other animals to restrict complex object recognition to small areas or objects at any one time that are in turn serially analyzed.

Two factors influence saliency: bottom-up and top-down. Bottom-up are stimulus-based memory-free factors that depend on instantaneous sensory input, while top-down are memory-bound goal-driven factors that consider the internal state of the organism, such as experiences or goals. A dramatic example of a bottom-up stimulus would be a fire cracker going off suddenly, while an example of top-down attention is the focusing onto difficult-to-find food items by an animal that is hungry, ignoring more 'salient' stimuli (Niebur 2010). Koch and Ullman (Koch and Ullman 1985) proposed the idea that the bottom-up visual saliency of a location is a combination of individual salient features (conspicuities). Subsequently Itti et al. (Itti, Koch, and Niebur 1998) provided a computational model for visual attention analysis. In the work presented in Chapter 5 we focus on the modulation of bottom-up saliencies based on this model.

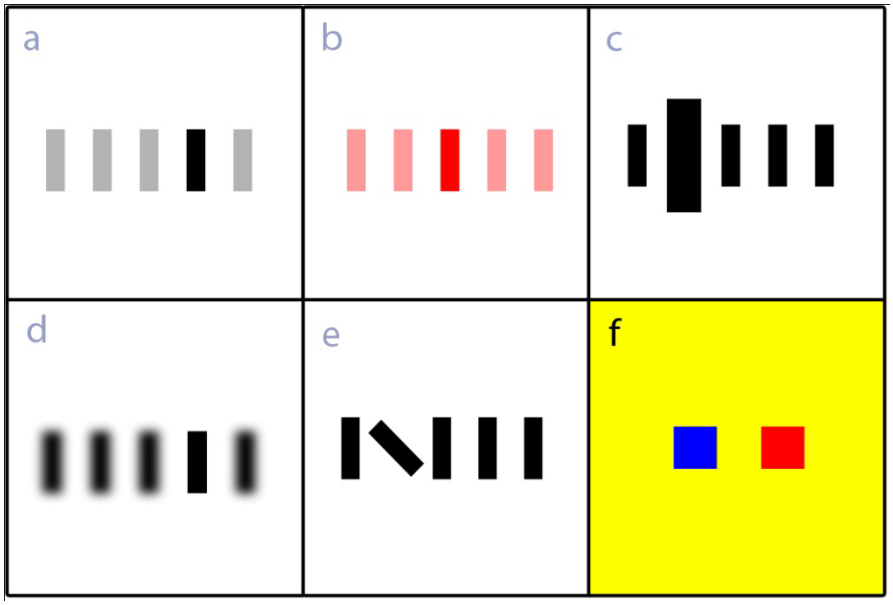


Figure 12. Examples of bottom-up stimuli. a) Lightness. b) Saturation. c) Size. d) Depth of Field. e) Orientation. f) Color-opponency.

There exist several dimensions of bottom-up stimuli. Figure 12 shows examples of lightness, saturation, size, depth of field, orientation, and color-opponents (also known as antagonistic colors). In each of these, one of the objects is more visually salient than the rest. However, opponent colors are less obvious; they are based on the opponent process theory (Hering 1964), which assumes that neurons are excited by one color in the center of their receptive field and inhibited by another, while the opposite is true in their surround (Itti, Koch, and Niebur 1998). Therefore, we perceive the colors by processing the differences between opponents. Chromatic color double opponents in humans are red-green, green-red, yellow-blue, and blue-yellow (Engel, Zhang, and Wandell 1997). This means that, for example, if a blue and a red object are placed on a yellow canvas, the blue will be more conspicuous due to it being a color-opponent of yellow (Figure 12, f). In Section 5.2 we use three bottom-up dimensions: lightness, saturation, and color-opponents. In Section 5.3 we use only two: lightness and color-opponents.

There is convergent evidence that there is a correlation between our visual attention and the saliency map. Ouerhani et al. used an eye tracker to compare the results of the saliency with actual tracked visual attention (Ouerhani et al. 2004). They presented a computational framework for assessing the plausibility of visual attention models. Also, they compared computational maps with human-eye–tracked maps that could receive multiple attention models. That work concluded that their initial studies suggested that indeed there exists a relationship between the computational model and human visual attention (see Figure 13).

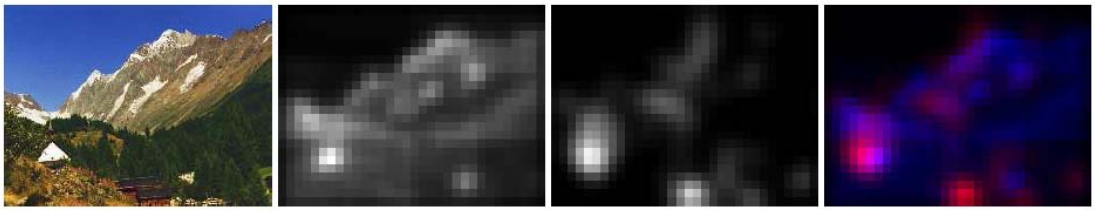


Figure 13. Comparison of a human eye tracked map and saliency map by Ouerhani et al. From left to right, color image, computational map, human map, comparison map. This particular example had a strong correlation ($\rho = 0.52$) (Ouerhani et al. 2004).

Figure 14. Example of eye tracking based NPR rendering. Information such as the license plate of the car is not abstracted while other distracting data (such as the background) is removed (Santella and DeCarlo 2004).

Santella et al. used also an eye tracker to evaluate the success of NPR imagery using saliency (Santella and DeCarlo 2004). The goal was to see what attracted the viewer's attention in abstract representation of images. They presented a user study that showed that the computational map of the saliency and the human attention behaved similarly in their ability to capture increased interest, but differed in their absolute capture of interest. Moreover, they also provided a mechanism to generate NPR images based on eye tracking. Figure 14 shows an example of their eye-tracking–based NPR image generation. Chapter 5 will also provide two eye-tracking user studies that suggest a relationship between the saliency map and our visual attention.

There is extensive work on trying to model the visual saliency of an image. The different techniques tried include non-parametric approaches, face detection or using trained samples over large datasets (Rosenholtz 1999; Cerf et al. 2008; Judd et al. 2009). However,

throughout this thesis, we will use the model proposed by the work of Itti et al., which will be described in more detail in Section 5.2.1 (Itti, Koch, and Niebur 1998).

2.4.3 Exploiting visual salience

Practically any change done to the image will modify its saliency map. Blurring, (de-) saturating, harmonizing and distorting are typical operations that implicitly change the saliency of the image. During the last few years, there has been an increasing interest in directing the attention of the user through saliency manipulation.

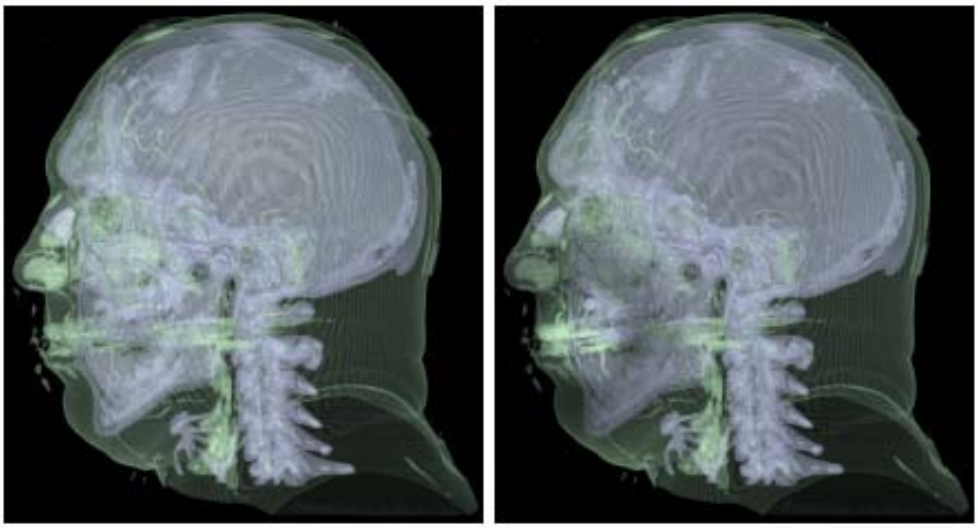


Figure 15. Examples of salience enhanced volume rendering. These two images illustrate how visual salience is manipulated when generating a volume render to direct the user towards the mouth of the dataset (Kim and Varshney 2006).

Kim and Varshney present a visual-saliency-based operator to enhance selected regions of a volume (Kim and Varshney 2006). They use this operator on a user-specified region to compute an emphasis field. They discuss how the emphasis field can be integrated into the visualization pipeline by modifying regional luminance and chrominance (see Figure 15). They also validate the effectiveness of their technique with an eye-tracker. Interestingly, they found that the saliency enhancement operator is more effective at eliciting viewer attention than the traditional Gaussian enhancement operator.

Two years later, the same group at University of Maryland applied saliency-based enhancements through geometry manipulation (Kim and Varshney 2008). They introduce geometry modification as a tool to direct visual attention. That work develops techniques to alter geometry to elicit greater visual attention (see Figure 16). Additionally, they also provide a user study based on eye-tracking to evaluate how successfully their technique guides user attention in a statistically significant manner. Their approach operates directly on geometry, and

therefore produces view-independent results that can be used with existing view-dependent techniques of visual persuasion (such as those presented in Chapter 5).

Figure 16. Visual salience modification by geometry. An example of how the geometry of an object can be modified to exploit visual salience, thereby directing the user's attention—on the right sphere, notice the patch on the upper right (Kim and Varshney 2008).

These works concentrate on creating salient features in focus regions, rather than applying subtle modifications to existing images. However, salience information can be used for other purposes. For example, Lee et al. used saliency to present a real-time framework for computationally tracking objects visually attended by the user while navigating in interactive virtual environments (Lee, Kim, and Choi 2007). Their framework not only uses bottom-up features, but also introduces goal-driven attention in order to better predict human eye gaze. Their framework first builds feature maps using bottom-up features such as luminance, hue, depth, size, and motion. The feature maps are then integrated into a single saliency map using the center-surround difference operation, based on Itti's model (Itti, Koch, and Niebur 1998). This pixel-level bottom-up saliency map is converted to an object-level saliency map using the item buffer. Finally, the top-down contexts are inferred from the user's spatial and temporal behaviors during interactive navigation and used to select the most plausibly attended object among candidates produced in the object saliency map. Moreover, they conducted a user study with an eye tracker to evaluate the prediction accuracy of their framework with respect to actual human gaze data. The accuracy level attained was well supported by the theory of human cognition for visually identifying single and multiple attentive targets, especially due to the addition of top-down contextual information.

Reducing geometric detail has been long a subject of research (Feiner 1985). Recently, Longhurst et al. used a technique to reduce the rendering fidelity of objects that are not being attended by users (Longhurst, Debattista, and Chalmers 2006) based on saliency computation. This is typically a computationally expensive technique, but their GPU based approach is capable of producing a "selective rendering" map quite fast. Interestingly, they use a few more attention dimensions: depth, habituation and motion. By doing this, they can

render their virtual scenes with high quality only in attended regions while using a significantly lower quality rendering for non-salient regions.

The closest work to the research presented in this dissertation (see Chapter 5 for more details) was carried out by Su et al. on de-emphasizing distracting image regions (Su, Durand, and Agrawala 2005). Bottom up saliency models that include, for example, luminance, color and orientation are called *first-order*. Su et al. focused on *second-order* saliency by considering texture variations. Once computed, these variations can be modulated to redirect the user's attention to specific locations. They tested their technique with 12 volunteers using an eye-tracker, and documented an impressive 22.43% speed-up on the mean response time, while at the same time being able to redirect the attention of the user. Highly relevant to our work (and similar to the findings of Kim and Varshney) is their finding that their texture equalization produced a stronger change in saliency than a Gaussian blur. However, their work only focused on the texture variation and did not consider the more straightforward first order bottom-up saliencies that are the focus of our research. Moreover, their system was not designed to run at real-time frame rates.

2.4.4 Other means of attention direction

Pixel-wise manipulation of an image is not the only way of directing people's attention. There are many techniques addressing this problem. Most of these techniques are augmenting, distorting, or modulating –also known as *in-place* (Kosara, Hauser, and Gresh 2003). One of the most widely known works of attention direction in AR was carried by Biocca et al. with the attention funnel. (Biocca et al. 2006). The omnidirectional *attention funnel* is a general purpose AR interface technique based on the *tunnel-in-the-sky* work of Barrows and Powel (Barrows and Powell 1999). The attention funnel rapidly guides attention to any object, person, or place in the space by augmenting the field of view of the user with a series of squares. Figure 17 provides screenshots from their application. In a study comparing the attention funnel to other attention-directing techniques such as highlighting and audio cueing, Biocca et al. found that the attention funnel increased search speed by over 50%, and decreased perceived cognitive load by 18%.

Using augmentations to draw the attention of users dates back several decades. For example, in 1993 Feiner et al. used a leader line to point to a desired destination, including destinations outside the view frustum (Feiner, MacIntyre, and Seligmann 1993). Also, Barrows and Powell presented in 1999 a cockpit display for aiding flight patterns (Barrows and Powell 1999). In fact, Barrows and Powell note that the development of the tunnel-in-the-sky display concept was started in the 1950s to improve situational awareness.

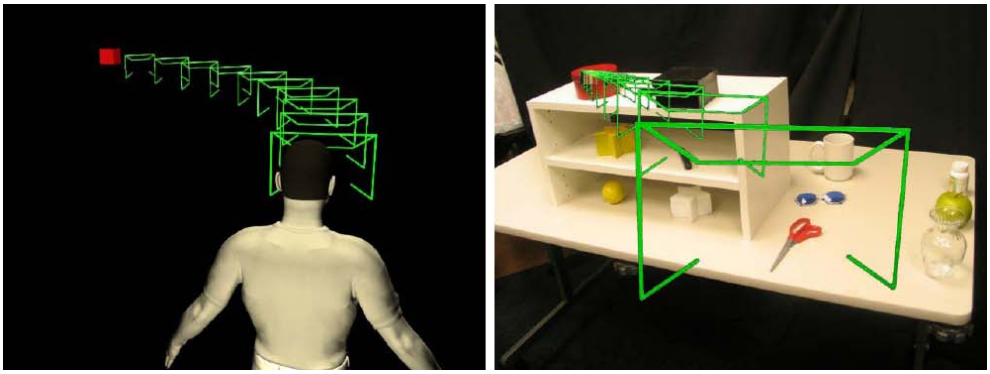


Figure 17. Attention Funnel. Examples of the attention funnel drawing the attention of the user to an object on the shelf, the red box. (Left) Virtual reality view. (Right) AR view (Biocca et al. 2006).

Also interesting is the research carried out by Bailey et al. on subtle gaze direction (Bailey et al. 2009). That work takes an entirely different direction than the one summarized earlier. Instead of augmenting or modifying the image entirely, they exploit the fact that human peripheral vision has very poor acuity compared to foveal vision. They then present a small stimulus on the regions of an image that currently has no foveal focus. Furthermore, they monitor saccadic velocity and exploit saccadic masking in order to remove the stimulus information before the user is able to explore it. They performed a user study with ten participants, using an eye tracker to evaluate the effectiveness of their technique. They found that they could influence the gaze direction of the participants but without reaching the modulated region. They attributed this to the undershooting of saccades (Collewijn, Erkelens, and Steinman 1988).

This chapter introduced work relevant to that done in this dissertation. The relevant work was organized in four sections. Section 2.1 introduced work that is relevant to the entire dissertation. Section 2.2 introduced work that is relevant to Chapter 3. Section 2.3 introduced work that is relevant to Chapter 4. And finally, Section2.4 introduced work that is relevant to Chapter 5. The articles cited here share core concepts with our work, though may differ on the actual implemented techniques. References to implementation details will be introduced in each chapter individually. We now begin the main chapters of this dissertation on the usage of context for AR. The next chapter uses contextual information to stylize portions of a scenegraph upon runtime.

Chapter 3

Using Context for Scenegraph Styling

This chapter focuses on using contextual information based on the presence of digitally tagged data. This contextual information is then used to influence the visual appearance of scene objects, be it by dynamically binding subgraphs, by influencing the actual generation of three-dimensional geometries, or by building a customized scenegraph.

The basic mechanism to achieve the abovementioned techniques is to maintain the contextual information as far as possible in the rendering pipeline. The work presented in this chapter is based on the concept of scenegraph traversals (Clark 1976) –see Section 2.1.2.1. All of the techniques rely on the assumption that there is a state being stored and forwarded to each node. This traversal state maintains the contextual information necessary for each node to use.

The first technique introduced in this chapter targets information filtering given contextual information. It is implemented based on the concept of magic lenses, as introduced by Bier et al. (Bier et al. 1993) –see Section 2.2.2. The presented technique used contextual information in combination with three-dimensional spatial regions to select a particular scenegraph before traversal. The resulting effect allows a single magic lens to affect the visual appearance of multiple objects differently. This behavior is unlike traditional magic lens strategies that apply the same effect to all objects that fall inside a lens' region of effect (Bier et al. 1993; Viega et al. 1996; Ropinski and Hinrichs 2004).

The second technique takes information filtering one step further by adding style mappings between subgraphs and current contextual states. Additionally, hierarchical aggregation of context is introduced and a clearer definition of the style and content counterparts is given.

The final technique acts on the lowest level of visual representation: the generation of geometries. In this technique, context-sensitive scenegraph traversal is coupled with an engine for procedural generation of models. The resulting technique can influence the way geometries are generated, by modifying generation parameters (such as level of detail) or even modifying the generation code itself.

Before we go into the details of every technique, the next section presents a short summary of context-sensitive scenegraph traversal. Context-sensitive traversal of scenegraphs was introduced by Reitmayr et al. (Reitmayr and Schmalstieg 2005). The techniques presented in this section build on that work.

3.1 Context-sensitive scenegraph traversal

In order for a scenegraph to be dependent on context, a set of context parameters is maintained throughout its traversal. This allows parameterizing and repurposing subgraphs in various ways. The context parameters are maintained as part of the state of the scenegraph traversal that makes them independent of the scenegraph structure. The context parameters are modeled as an associative array of key-value pairs, where the values are either strings or pointers to subgraphs.

By using pointers as parameters, such template subgraphs can be inserted multiple times during the traversal. In contrast to a conventional directed acyclic graph structure, the binding of child nodes to their parents happens very late, during the traversal itself, so the nodes can be changed for each traversal and provide a very flexible way of assembling complex scenegraphs. Transparent caching of the traversal outcome in display lists ensures that rendering performance is not adversely affected.

The context-sensitive scenegraph traversal shifts the complexity of managing multiple representations from the application code to the scenegraph itself. Rather than writing application code to modify the scenegraph or change rendering parameters based on user interaction, the application only needs to change the context parameters for high level control of the visual effects.

The scenegraph adapts the visual appearance of its contained objects given dynamically changing requirements and it even may compose subgraphs on the fly. A particular visual representation is simply an instance of a template subgraph combined with a specific choice of context parameters. Combinations of content and visual interpretation are created during traversal only at the actual moment in time at which they are required.

3.2 Using context for information filtering

The general idea of information filtering is to modify the appearance of virtual scene objects based on user-defined context parameters, typically fading out uninteresting scene objects to reduce display clutter. *Magic lenses* are filters that modify the presentation of scene objects in a locally bounded area. They can be used to reveal hidden information, to enhance data of interest, or to suppress distracting information (see Section 2.2.2).

In the original definition of magic lenses, the effect of a lens is locally bounded, although it is applied globally to all scene objects. The effect of multiple lenses can be aggregated by overlapping multiple lenses, but this does not allow applying the effects only to certain individual scene objects or groups of scene objects. For example, a composite effect of enlarging all scene objects of type *A* by 10%, while rendering all scene objects of type B semi-

transparently, cannot be composed from a lens that enlarges everything by 10% and a lens that renders everything semi-transparently.

In this section, we introduce the notion of three-dimensional context-sensitive magic lenses (CSMLs) that overcome these deficiencies. Every scene object's rendering style is defined as a function of an arbitrary set of context parameters. These parameters are contained in a group with other arbitrary information, such as object type or membership in a certain branch of the scenegraph. The position and extent of the magic lenses, the type and arrangement of objects in the scenegraph, and the mapping from context parameters to rendering style are completely independent, allowing mixing and matching tools from this toolset, even at runtime—the only requirement is that all used components agree on a common set of context parameter descriptors.

In general terms, the main contribution of CSMLs is that they provide powerful information filtering for arbitrary complex scenes, without most of the limiting assumptions of previous work in that area. Specifically, these lenses are completely general in terms of their shape, number, and effect on the visualization. Although designed for AR, the same magic lens techniques can be applied to purely virtual scenes. This technique is designed with outdoor AR applications in mind that draw from a rich database of virtual objects associated with real world coordinates and entities, such as those of Geospatial Information Systems (GIS). Therefore, hard-coded visualization behaviors are not a satisfactory option.

We now describe the CSML based on the scenegraph parameterization. For rendering, we use a modified version of the algorithm by Ropinski and Hinrichs (Ropinski and Hinrichs 2004).

3.2.1 Background: 3D magic lens rendering algorithm

The algorithm described by Ropinski et al. consists of three rendering passes. Algorithm 1 outlines the procedure.

```
Render fragments behind and next to the lens.
Render fragments inside the lens.
Render fragments in front of and next to the lens.
```

Algorithm 1. 3D Magic Lens Rendering Algorithm. This algorithm allows the rendering of three-dimensional volumetric magic lenses in three passes. (Ropinski and Hinrichs 2004).

Note that every object that we want to be affected by the lens must be rendered up to three times. The first and second passes require two depth tests for distinguishing the fragments falling behind and inside the lens, respectively. In the second pass, the style changes are ap-

plied. Figure 18 shows a conceptual representation of the space partitioning of this algorithm.

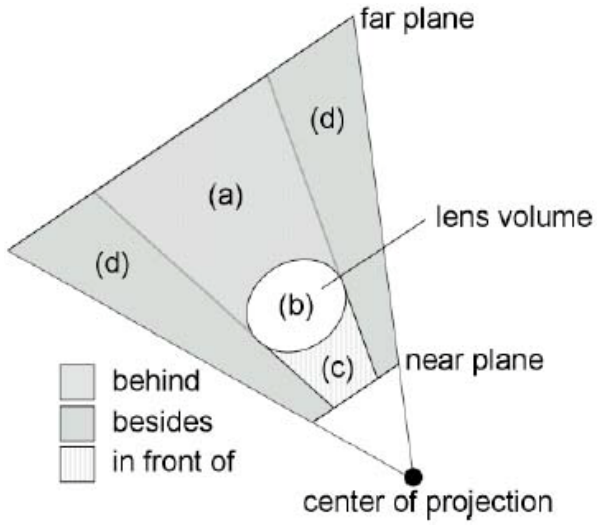


Figure 18. Algorithm by Ropinski et al. This figure illustrates the subdivision of the view frustum into three sections – (a) behind the lens, (b) inside the lens, (c) in front of the lens, (d) remaining fragments (Ropinski and Hinrichs 2004).

3.2.2 Modifying the algorithm

Instead of using a shadow or stencil buffer, we rely on Cg fragment programs (Mark et al. 1993) combined with floating point textures to overcome the lack of two depth tests. This means that the magic lens will have one texture associated that stores its depth information. In subsequent rendering passes, this texture will be used by all other objects to determine if they fall inside the lens or not. A more formal description of the algorithm is given on Algorithm 2.

It must be noted that this algorithm does not consider any contextual information attached to the objects. This is because contextual information is added as a part of a subgraph of the scene during traversal, as will be explained in the following section.

3.2.3 Implementation

We provide the details of an implementation for multiple magic lenses. A *context family* of objects is defined by all objects that have the same context information. However, these objects do not have to belong to the same part of the scenegraph hierarchy (i.e., they can be scattered throughout the scenegraph and still be jointly affected by a CSML according to the context family to which they belong). Every context family has a unique name and defines a specific rendering style that will be used when fragments of its objects are inside or outside a CSML.

```
Let t be the texture that will hold the depth information of the
magic lens l
Let c be a string representing the context information
Let o be a group of 3D objects
Let s_j={o,c} be the j^th group composed of a group of objects and their
context information, where j=0…m

If l is encountered in the graph:
     Render it such that the depth values of its back faces are
     stored in the red channel of t

     Render it such that the depth values of its front faces are
     stored in the green channel of t
End if

For every s_j encountered in the graph do
     Render it using the depth information stored in t to display on-
     ly those fragments lying behind and next to l

     Render it using the depth information stored in t to display on-
     ly those fragments lying inside l

     Render it using the depth information stored in t to display on-
     ly those fragments lying in front and next to l
End for
```

Algorithm 2. 3D Magic lens rendering on the GPU. This algorithm is a modified version of that given by Ropinski and Hinrichs. It renders objects inside the lens by using textures to support the depth information.

We created three new scenegraph nodes for these purposes:

- *SoCSMLFamilies* that sets the appropriate rendering style for every context family.
- *SoCSMLLens* that holds the convex shape that defines the region of our magic lens.
- *SoCSMLScene* that holds a group of objects and the context family to which they belong.

In the following subsection, we give a more detailed description of these nodes, based on their scripting interface. The code snippets in this chapter will be given in Open Inventor Format (Wernecke 1994). Throughout this dissertation we will use the implementation of the Open Inventor framework by the company Coin3D (SIM 2010).

3.2.3.1 SoCSMLFamilies

This node allows the definition of the rendering styles for the context families of objects. An example of its declaration is:

```
SoCSMLFamilies {
      Family "alpha"
      lensName [ "RedPainter", "BluePainter" ]
      style [ USE Red, USE Blue ]
}
```

Here, all the pixels of the objects of the family “alpha” will be rendered in red when they are inside the magic lens called "RedPainter", whereas all those pixels of the same family that fall inside "BluePainter" will be rendered as blue. The style definitions referred to by the “USE” keyword are actually references to subgraphs of arbitrary complexity, containing any form of rendering effect possible in Open Inventor.

3.2.3.2 SoCSMLLens

This node accepts an arbitrary subgraph as a geometric description of the magic lens. For example:

```
SoCSMLLens {
      Name "RedPainter"
      Content Sphere { radius 2 }
}
```

Here, a sphere of radius 2 will act as a magic lens, and the name assigned to this lens is "RedPainter". Notice that the geometry of the lens must be a convex polyhedron, but can be of an arbitrary complexity.

3.2.3.3 SoCSMLScene

This node accepts an arbitrary subgraph as content and a string that acts as the contextual information. When scenegraph traversal happens, this node builds a scenegraph on the fly with the rendering styles defined for this context family for the pixels that lie inside and outside the lens. For example:

```
SoCSMLScene {
      Family "alpha"
      Content Cube {}
}
```

Here, a single cube is the sole part of our CSML Scene and has been assigned as part of the “alpha” family. Notice also, that, although a single shape is assigned as the content here, this can actually be a more complex subgraph.

Conceptually, the magic lenses affect the rendering style of the objects, but in practice, it is the objects inside a CSML Scene that determine the family to which they belong, enabling them to build the subgraph according to their contextual information. Ultimately, the magic lenses only define the regions where the rendering styles are changed.

For efficiency issues, we use frame buffer objects (Juliano and Sandmel 2006) for sharing of the texture between the magic lens and the group of objects with contextual information. To reduce aliasing artifacts, we use floating-point textures to store the regions of the lenses. Other techniques, such as supersampling, may be employed to further reduce artifacts.

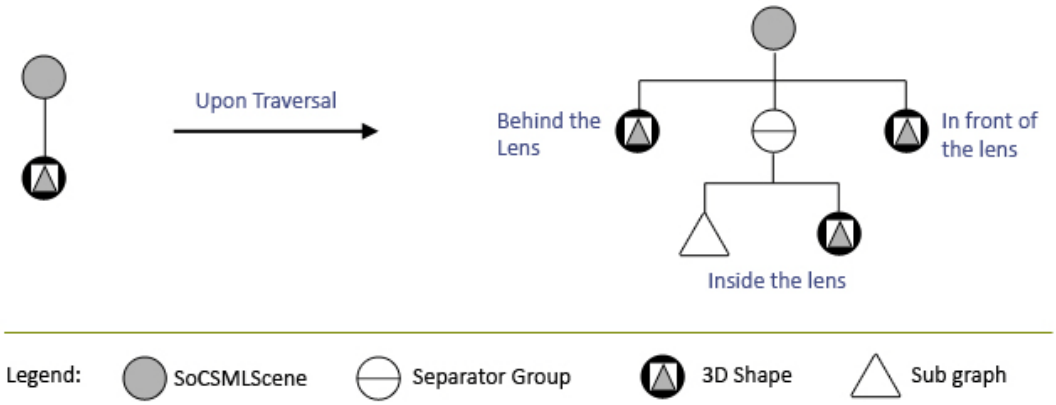


Figure 19. Overview of how a CSML Scene is built from given content. Upon traversal, the content is traversed three times. Every style subgraph (triangle) represents the information defined by the context family.

The multiple passes required by every CSML Scene are achieved by building a subgraph with multiple references to the content subgraph to be rendered. In those passes in which specific styles must be applied to pixels falling inside the lens, we build the subgraph with the information associated with the respective context family. Figure 19 illustrates how a graph including a CSML is built during traversal. The parameterization of the subgraphs is based on the context sensitivity mechanism outlined by Reitmayr et al. (Reitmayr and Schmalstieg 2005) described previously. Note that the subgraph is referenced multiple times, independent of the number of objects inside it. This subgraph is constructed with the sequential traversal of the style subgraph (defined by the CSML Families) and the content subgraph (given to the CSML Scene). The style subgraph precedes the content subgraph and can therefore influence its appearance. In the case that multiple magic lenses are in the graph, the same content subgraph will be rendered with different style subgraphs. In this sense, the style subgraph that is provided separately can be seen as a style parameter to CSML rendering.

It is important to note that the style parameters are completely arbitrary subgraphs, and can incorporate any standard or user defined features of the scenegraph. The styles are thus not constrained to simple color or transparency changes.

The example scenegraph shown in Figure 20 illustrates how the context families do not need to be hierarchically grouped, and yet, because of context sensitivity, they are correctly affected by the magic lens. Figure 21 shows a 2D conceptual view of how context sensitivity can affect objects in a complex scene. In this illustration, the pixels of those objects that fall inside the lens (the grey semi-transparent square) are rendered with a specific style parameter, depending on the context family to which they belong. The objects in the image are not grouped in subgraphs, but they are grouped by context.

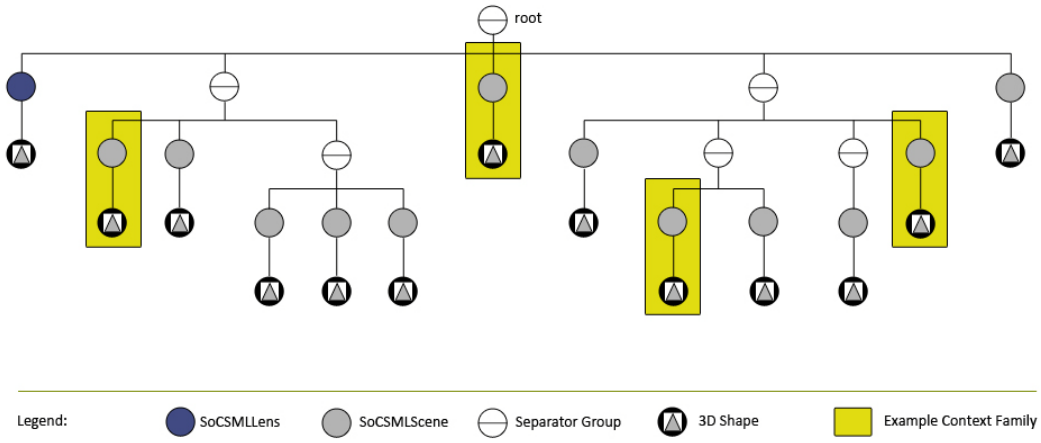


Figure 20. Illustration of context families. Example scenegraph illustrating how the context families do not need to be hierarchically grouped and yet are jointly affected by the lens given their context information.

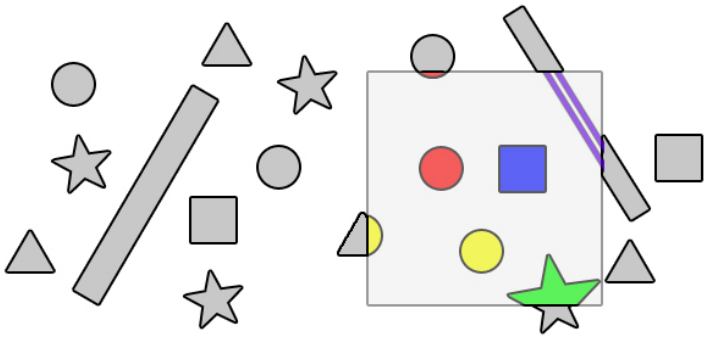


Figure 21. CSML concept. A conceptual 2D view of how a CSML affects a complex scene. All fragments intersected by the lens are rendered differently, regardless of their position in the graph, affecting not only the rendering style, but all aspects that can be composed as a sub-graph.

Transparency always presents issues in interactive applications. Several techniques have been developed to overcome this problem, such as the work of Everitt Cass on order-independent transparency (Everitt 2001). While our technique can efficiently make use of, for example, alpha blending with one lens in the scene, the behavior turns more complex when multiple lenses come into play. This is because we do not restrain the position of the lenses in our graph and they can be mixed in the hierarchy of the scene. To solve this problem, we require a small modification of the last rendering pass, to allow the objects in the scene to check whether they are effectively outside every lens area (given by the textures).

This technique may decrease the rendering speed only slightly because no extra rendering passes are necessary. However, it can be deactivated by user request. Also, it must be noted that, when this technique is used, the last pass checks the effective outside regions of all the lenses, and consequently, the first rendering pass becomes redundant and can be deacti-

vated (by the removal of the first branch of the subgraph). Since the rendering order of the lenses is controlled by the user, this gives us the power to control the order of the style parameters in a scenario where multiple lenses intersect. However, when it comes to transparent content inside a lens, an arbitrary lens order forces us to use an order-independent rendering strategy such as screen door.

3.2.4 Results and applications directions

We have envisioned a number of applications that can be addressed by the use of CSML. In the following, some practical examples are outlined.

3.2.4.1 Modeling of an X-ray vision frustum

An obvious use for magic lenses is that of X-ray vision. Bane and Hoellerer (Bane and Hoellerer 2004) developed an interaction tool for X-ray vision that used two approaches: volume and room based. In particular, the volume approach can be addressed by the use of CSML. The authors describe a tool that creates a virtual frustum in the field of view of the user (called "tunnel tool"). The rendering style of the objects is then affected if they intersect this virtual frustum.

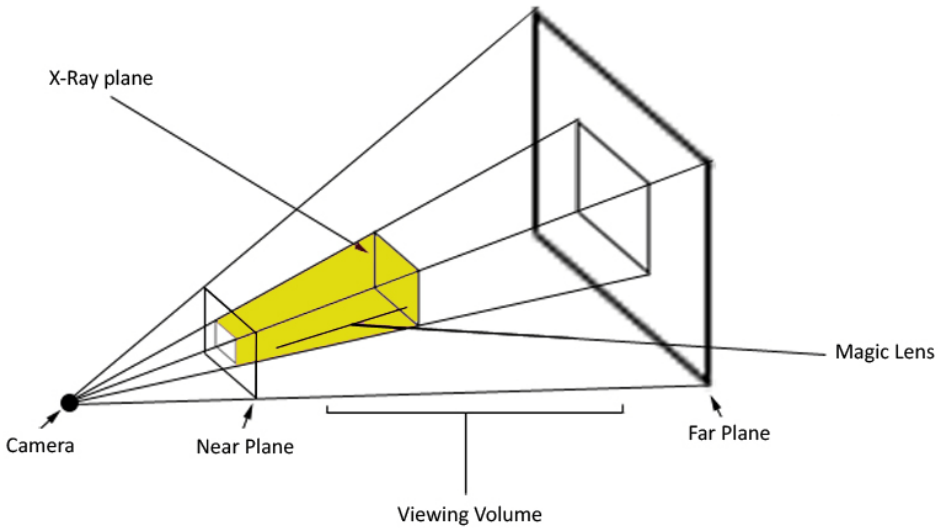


Figure 22. X-ray tool concept. A conceptual diagram of how a single CSML can be used to mimic an X-ray vision frustum.

This work uses the concept of layers, where objects can belong to a "class" that determines whether an object is displayed or not. However, it appears that objects are not affected differently when they intersect the frustum. In other words, the context given to the objects by the use of the layers does not include the intersection with the lens.

We have designed an approach similar to that of the X-ray frustum that makes use of CSML. Figure 22 presents a conceptual diagram of how such a lens is used. One scenario for this

concept is that all objects falling inside the lens are made semi-transparent by adding them all to the same context family. However, to truly exploit context sensitivity, objects should be affected depending on their context family. For example, the outside walls of a building that fall inside the lens will be made semi-transparent, whereas interior objects, such as furniture, are rendered in a standard color (blue), while objects of higher importance, such as characters, are highlighted (red). As the camera moves or the *X-ray* frustum moves, those pixels of the objects falling inside the lens are affected accordingly. Figure 23 (left) shows a screenshot from an application of this concept.

3.2.4.2 Context-sensitive X-ray tool

Naturally, our X-ray tool does not have to be attached to the viewing frustum and can be moved and adjusted interactively by the user. Figure 23 (right) shows a user holding a tracked magic lens in front of a liver model prop. By intersecting the lens with the liver, the user is able to see the vessel trees inside the liver. In particular, the vessel trees are differently colored, in red or blue, while the parenchyma of the liver that falls inside the lens, is made transparent.

3.2.4.3 Localized visual disambiguation

In a dense AR scene, augmentations can often be ambiguous. Such a case occurs in Figure 24 (top), where multiple families of objects with different semantics are represented in a similar style. Gas and electricity pipes have been rendered on top of the video input. Given the geometrical similarity of both types of objects, it is impossible to tell them apart. A common technique to disambiguate these objects is to change their color (middle), or perform some other type of rendering style change. Conventionally, such color coding is applied to the entire scene. CSML offers the opportunity to selectively apply the disambiguation. Users can place a lens in a specific area (Figure 24 bottom) where they want to disambiguate objects, without affecting the entire scene.

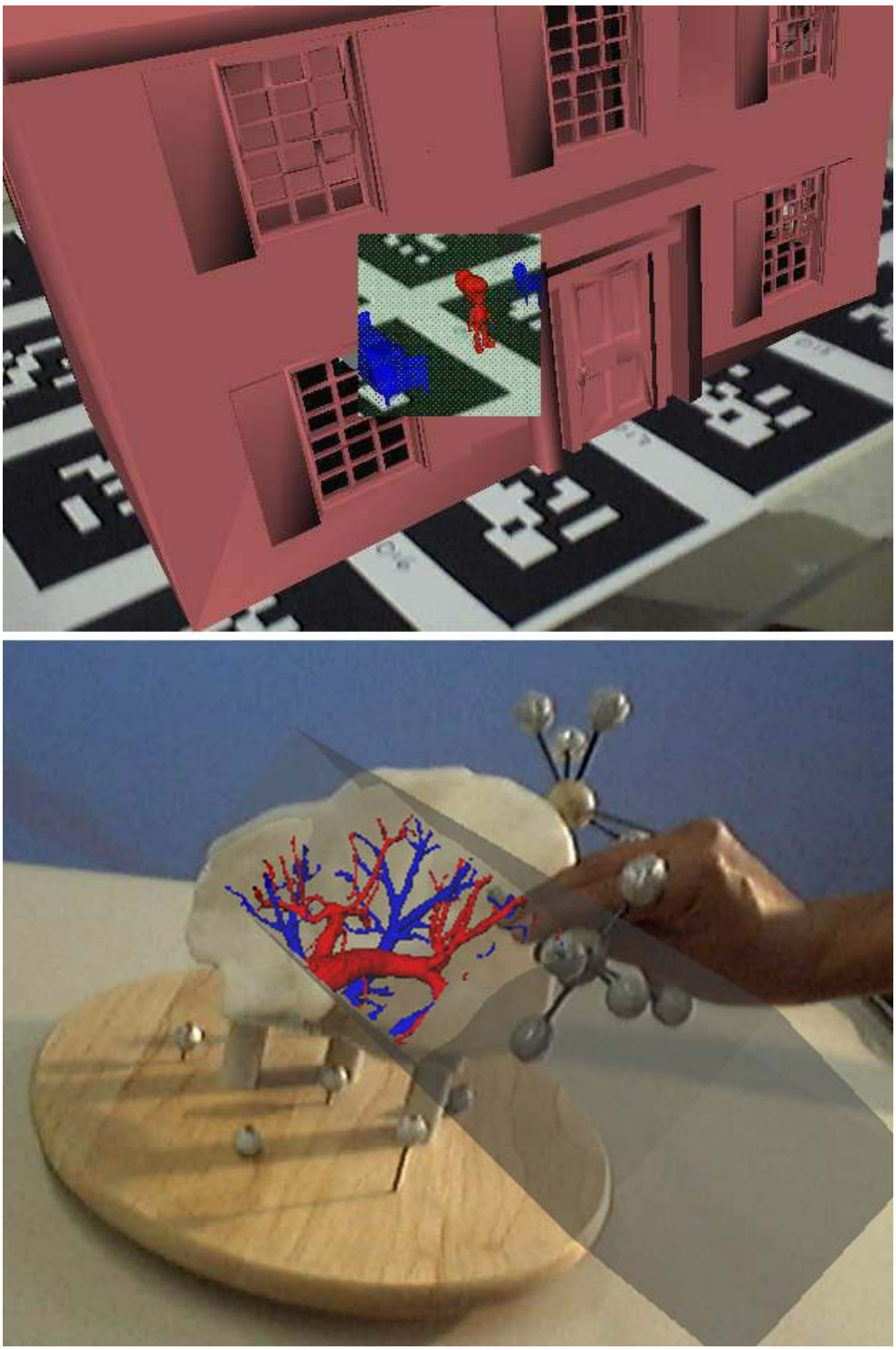


Figure 23. Screenshots of X-ray applications. (Left) An X-ray vision frustum-like technique implemented with CSML attached to the camera. (Right) A handheld X-ray vision tool.

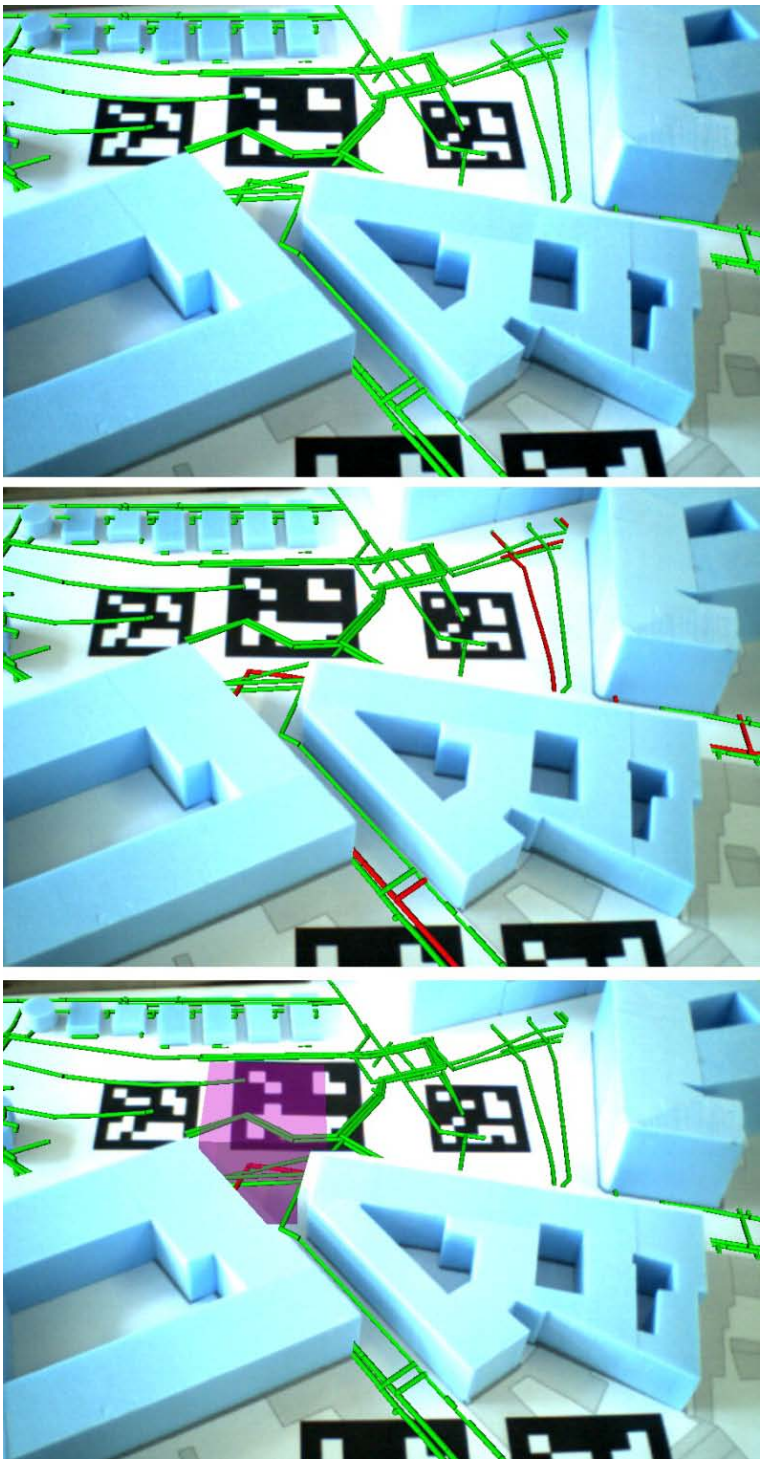


Figure 24. Localized visual disambiguation. (Top) A scaled model of downtown Graz is overlaid with power lines and gas pipe. As can be seen, it is an ambiguous representation. (Middle) Gas pipes have been colored red throughout the scene. (Bottom) The style rendering change has been localized to be inside the lens geometry. This is possible with a context rich scenegraph. The blue foam models were created from real geographical data at a scale of 1:250. The power and gas data come from the same dataset.

3.2.4.4 Cutaway volumes and ghosting

Feiner and Seligmann discussed a number of techniques to satisfy visibility constraints in dynamic three-dimensional illustrations (Feiner and Seligmann 1992): object removal, ghosting, and cutaways. All of these styles can be modeled by CSML.

Figure 23 shows an example of ghosting, while Figure 25 shows an example of cutaways. We define four families of objects, non-occluding, ghosts, cutaways and occluding. Those objects belonging to the family of ghosts will be made semi-transparent when they fall in the line of sight between a non-occluding object and the camera.

Similarly, objects belonging to the family of cutaways will not be displayed and occluding objects will be shown without any rendering style modification. This can be achieved by dynamically creating a lens that is formed from the back faces of the non-occluding object and the near plane of the viewing frustum. This design is similar to the concept of the X-ray vision tunnel presented before.

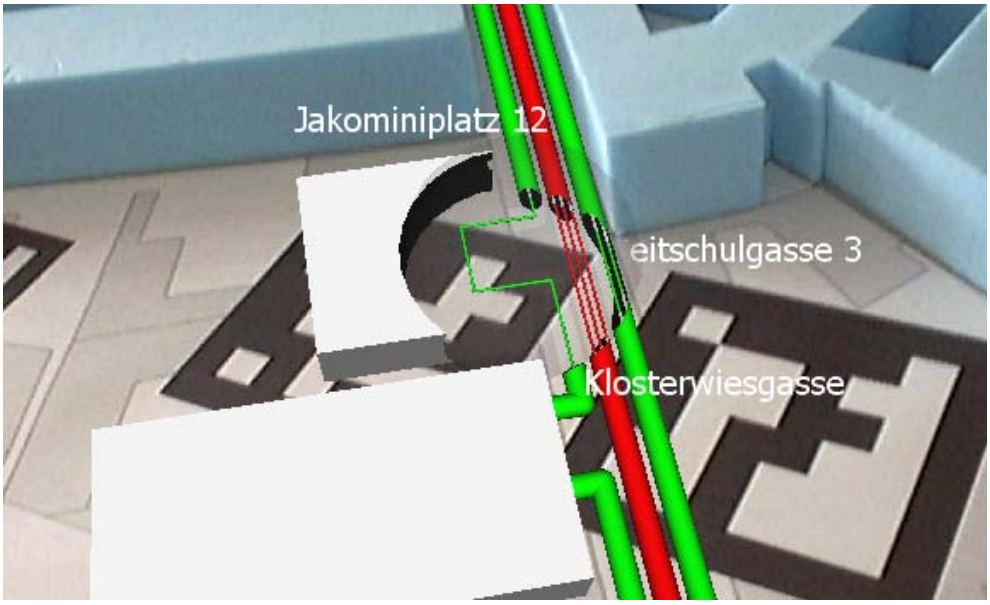


Figure 25. Information revealing. A CSML is used to enhance the information of some objects (details on pipes structures) and to present hidden information (by object cutaway of the buildings).

3.2.4.5 Information revealing

As described by Bier et al., two more possible uses of magic lenses are to enhance data of interest and to reveal hidden information. These two behaviors can simultaneously be achieved with CSML.

For example, Figure 25 shows a real model overlaid with three-dimensional representations of buildings; between them power lines (red) and gas pipes (green) are presented (in this

case, for illustration purposes, we use mockups.) A lens has been placed in the scene, intersecting the pipes and the virtual buildings. Those pixels of the buildings that fall inside the lens are shown using a cutaway view, revealing the information behind them. In turn, the pipes are enhanced with more detailed information on their structure.

3.2.5 Discussion

The idea of CSML came from working with highly complex AR scenes such as geo-data models, where global information filtering does either too little or too much for the desired effect. The heterogeneity of our complex scenegraphs precluded the use of traditional three-dimensional lenses, because every special case for every object family would have to be hard coded. As a solution, we have introduced context-sensitive behavior for magic lenses to enhance usability and rapid prototyping of interactive three-dimensional environments.

The use of context sensitivity in a scenegraph permits dynamic creation of subgraphs during traversal (Reitmayr and Schmalstieg 2005). While this makes scenegraph design more complex, it can mostly be overcome in larger practical applications by automating the generation of scenegraphs through translators and script generators, rather than handcrafting the data. This line of thought has been further explored and is presented in Section 3.4.

This section presented the first attempt at using tagged contextual information to influence the visual appearance of scene objects. It does so by combining the concept of context-sensitive scenegraph traversal with the idea of magic lenses. The final result allows application designers to create dynamic visual results without hard coding the data, but depending on contextual information. The next section will push the idea of CSML and further provide a generalization of the matching mechanism between styles and context families.

3.3 Style mapping and hierarchical aggregation of context

The previously discussed technique requires matching the context families and the styles that affect them. This matching has the limitation that it is given manually, but this does not have to be so. The technique presented in this section does not have that shortcoming. It is based on defining context for scene objects and style maps for visualization styles, and deriving the actual style for an object on the fly as a function of both context and mapping. Both context and maps are hierarchically organized and can be controlled independently.

Ideally, application behavior leading to the definition of the styles, the selection of styles, and the objects to which the styles are applied, should be separate concerns that can be designed and maintained independently. The application code should only control the mapping of styles to objects fitting a particular characterization, without having to touch the objects directly. In this way, a data driven application becomes feasible that allows content, styles and

behavior to be varied independently in order to address a wide range of possible applications using the same AR system. This separation relies on two simple techniques:

- **Context markup** is a technique of attaching freeform context attributes to nodes in the scenegraph hierarchy. The context is propagated downwards, and can be aggregated or modified, as more context definitions are encountered during traversal. The context can either be user-defined or derived automatically (e.g., based on uncertainty estimation).
- **Style templates and maps** define a mapping of context markup to style templates. This mapping can be defined directly in the scenegraph or attached from outside to the root of the scenegraph. The latter approach allows defining and manipulating the style template independently of the scenegraph. Style maps are organized hierarchically, similar to cascading style sheets for XHTML.

The binding of styles to objects happens very late, just before the traversal of the objects. Using context to connect styles to objects allows a clean separation of concerns and an arbitrary mix of local and global control of style. In particular, any procedural code that selects appropriate styles need not know details of the type, number or location of objects in the scenegraph. The approach works particularly well when dynamic variation of styles is needed, such as in applications letting the user pick objects of interest interactively, for example, as in visualization techniques such as focus + context. This section shows that contextual information may be exploited for defining visual styles for AR with little knowledge of the scene to be displayed.

3.3.1.1 Context markup

A *context family* of objects is defined as all those objects that have the same contextual markup. However, these objects do not have to belong to the same part of the scenegraph hierarchy—they can be scattered throughout the scenegraph and still be jointly affected according to the context family to which they belong.

An important consideration is that, unlike the work described in Section 3.2, the membership of an object in a particular context family is not explicitly given, but is the result of an aggregation of context attributes encountered while traversing the path from the scenegraph's root node to the object itself. This mechanism works exactly like other traversal states such as transformations, where these are aggregated depending on their position in the graph. Context attribute nodes defining arbitrary key/value pairs are inserted as markups at arbitrary positions of the scenegraph. Any context attribute encountered during the traversal is added to a set of current context attributes maintained during the traversal. Attributes can be overridden during the traversal if a second context node specifying the same attribute key is present.

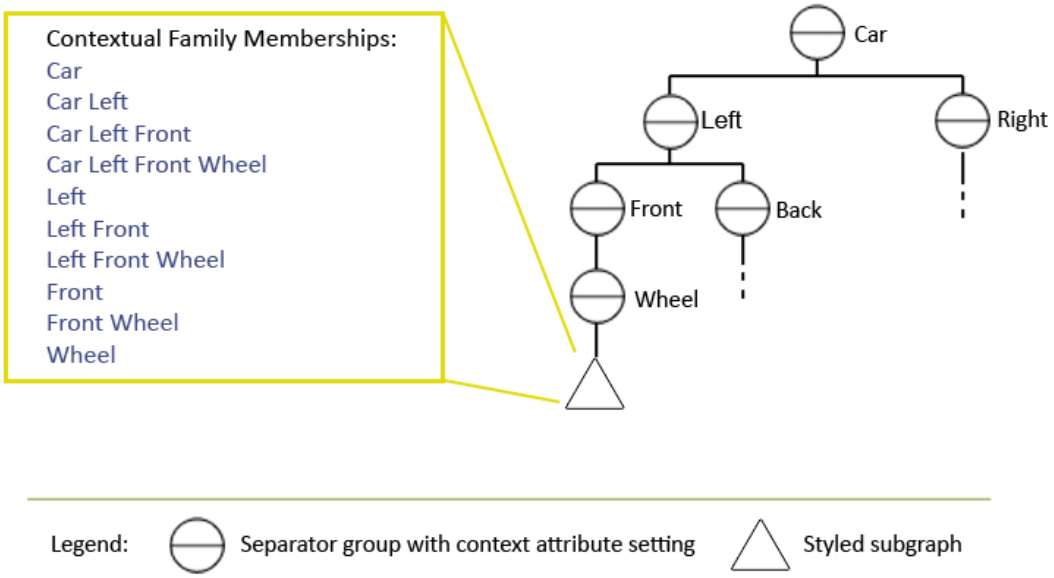


Figure 26. Illustration of the contextual family memberships of a styled subgraph. The memberships are all possible combinations of individual context attributes present during traversal. The styled subgraph may be referenced by any of the combinations of individual context attributes present during traversal.

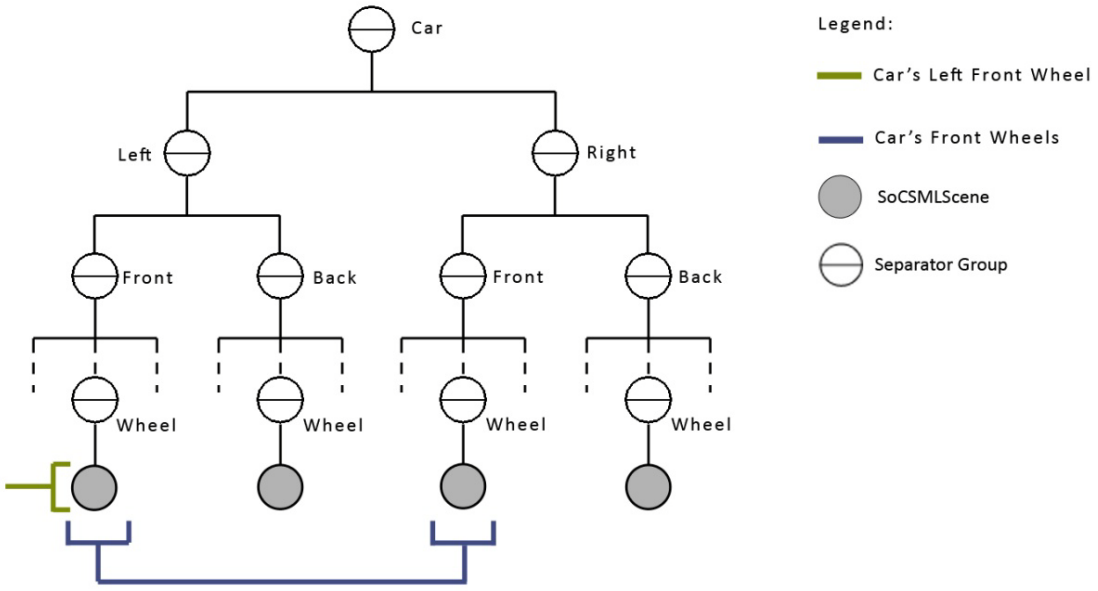


Figure 27. Context family referencing. An illustration of how a slight difference in the context values used for referencing may address objects in hierarchically separated portions of the scenegraph. If we reference only the car's left front wheel (green) we get an isolated node. However if we remove the "left" restriction, the objects whose context values match are in separated regions of the graph (blue).

The context family of a particular subgraph—marked up as a *styled subgraph*—is defined by the set of available contextual attributes at the moment of the subgraph's traversal. It is important to notice that every styled subgraph is simultaneously a member of several context families. This is because every context family may be composed of any combination of the partial or total context attributes present during traversal. We illustrate this in Figure 26 with the diagram of a scenegraph and the context families of which the styled subgraph (grey circle) is a member. Membership of the different context families is not known until the actual traversal, since context values may be dynamically changing.

Furthermore, a simple change in the combination of context values used to reference a family may target hierarchically disconnected regions of the same scenegraph. Figure 27 illustrates this situation with a conceptual representation of a scenegraph. The red highlight illustrates a reference to objects belonging to the family "Car Left Front Wheel". Notice that this is a very restrictive combination of context values and it consequently yields only one object. If we were to remove one of the restrictions, namely "Left", this would reference objects in hierarchically separated parts of the graph (highlighted in blue). Traditionally, this mode of referencing would either demand a detailed knowledge of the scenegraph or require expensive search actions. However, this is easily achieved with our context markup, since every object can retrieve its family memberships from the traversal state.

3.3.1.2 Style templates and maps

A style template is a system resource given in the form of a named styling subgraph. This is typically composed of nodes controlling the visual appearance such as material definitions, textures, GPU shaders or even transformations. By allowing arbitrary scenegraphs as style templates, we target the widest possible control of visualization styles, including custom nodes and side effects. Style templates are used to influence the appearance of styled subgraphs. During the rendering traversal, every styled subgraph is preceded by the first style template mapped to one of its context families.

The mapping of a context family to a style template is determined by another system resource, the style map. Every entry in the style map assigns a particular context family—given by a set of context attribute key/value pairs—to a style template. The style map is composed of individual nodes, each defining one mapping arranged hierarchically as part of the scenegraph.

It is recommended that the content portion of the scenegraph—containing the styled subgraphs marked up with context attributes—be kept in separate regions of the graph from the style templates and style maps.

Nonetheless, the nodes of the style map are arranged hierarchically and their definition is extracted by traversal. This means that hierarchical styling rules can be constructed by defining more specific styles first, and then progressing to more general styles until a final default style with empty context. During traversal, every styled subgraph will check the defined mappings and then, depending on whether its own context markup is mapped to a template, it will fetch the appropriate styling subgraph. This is illustrated in Figure 28. Notice that the binding of rendering styles happens during traversal; therefore, we neither need to specify a style for every object, nor need to know the data structure arrangement of the scene. It must also be mentioned that the search for matching mappings does not depend on the combinatorial number of context family memberships of the styled subgraph, but on the linear number of available mappings.

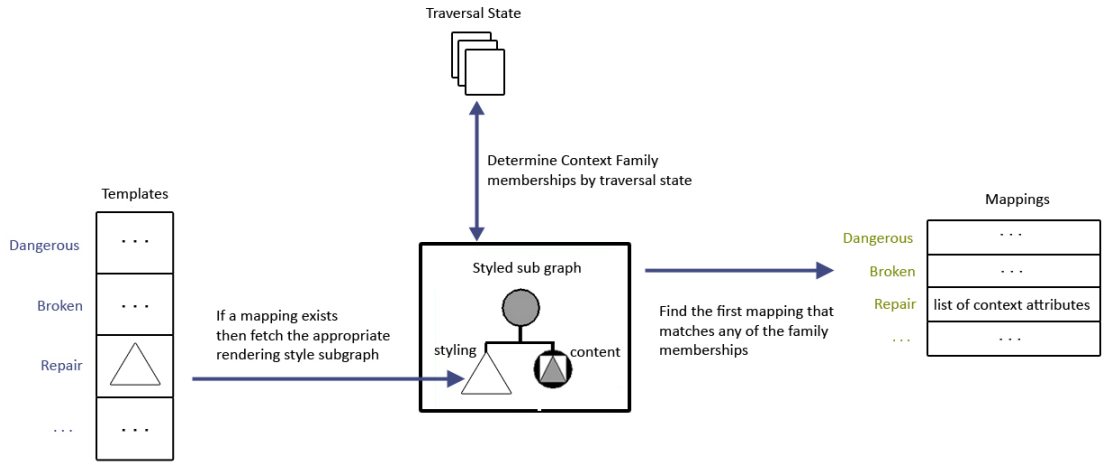


Figure 28. Template mapping upon traversal. During traversal, the styled subgraphs check for the first mapping that matches their current context markup. If so, the appropriate styling subgraph is appended immediately behind the content of the styled subgraph. Nested styled subgraphs are allowed and therefore incremental styling is possible.

The typical overall scenegraph structure consists of three subgraphs traversed in the following order:

1. Subgraph containing all style templates
2. Subgraph containing all style maps
3. Set of subgraphs containing context attribute nodes and styled objects, called styled subgraphs

However, it is also permissible to interweave these three aspects at the expense of increased scene management complexity. The separation into distinct subgraphs mainly serves clarity and separation of concerns. In particular, all three aspects can be developed independently and by different designers, as long as the style names and context attributes are consistent.

Figure 29 shows a conceptual diagram of the linking among style templates, context markup and style mappings. The hierarchical nature of the context markup allows the mappings to reference any context family available in the scene. In turn, the mappings function like cascading style sheets, since every styled subgraph will progressively search for the first mapping that matches one of its context families. It is important to remark that the contents of a styled subgraph do not necessarily have to be atomic three-dimensional objects, but can in turn be complex subgraphs or even nested styled subgraphs.

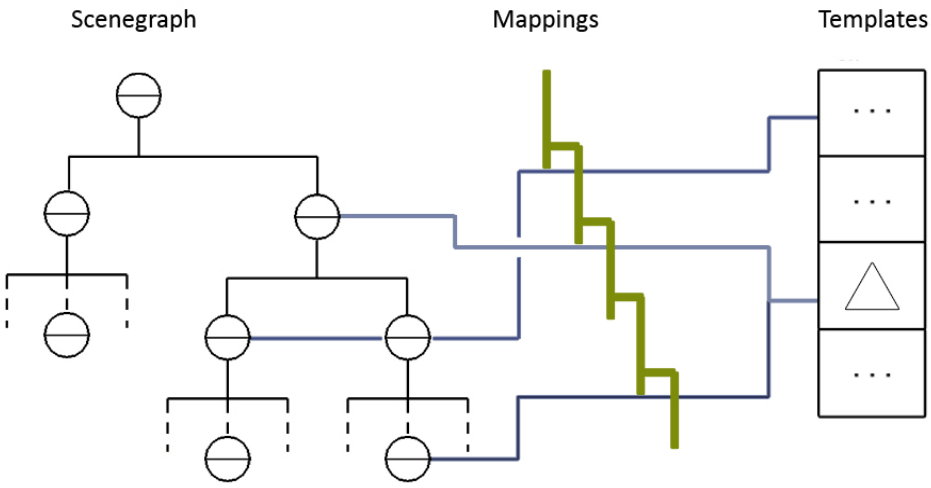


Figure 29. Conceptual mapping. This diagram shows how templates and context markup are linked together by mappings. Notice that the hierarchical nature of the mappings allows them to work similar to cascading style sheets. However, the separation of style and map allows reusing templates given different mappings.

All four building blocks (style templates, style maps, styled subgraphs and context markup) may be defined by different sources, ranging from user selection to automatic generation given higher rule sets. Figure 30 shows a diagram of possible sources, for example, styled subgraphs may come from legacy data bases such as those from the GIS community. GIS data is traditionally enriched with contextual information that may also serve as input for the context markup and this concept is further explored in Section 3.4. Another example are tracking uncertainties like those described by Coelho et al. (Coelho, MacIntyre, and Julier 2004), which may also be used as context attributes. The style templates, however, are usually defined by application designers, since visual styles and naming conventions may carry a semantic meaning (e.g., the generation of a style for “Danger” is not easily generated automatically). Style maps, in turn, may come from a range of possible sources, such as high complexity rule sets that consider user properties and tasks (Julier et al. 2002).

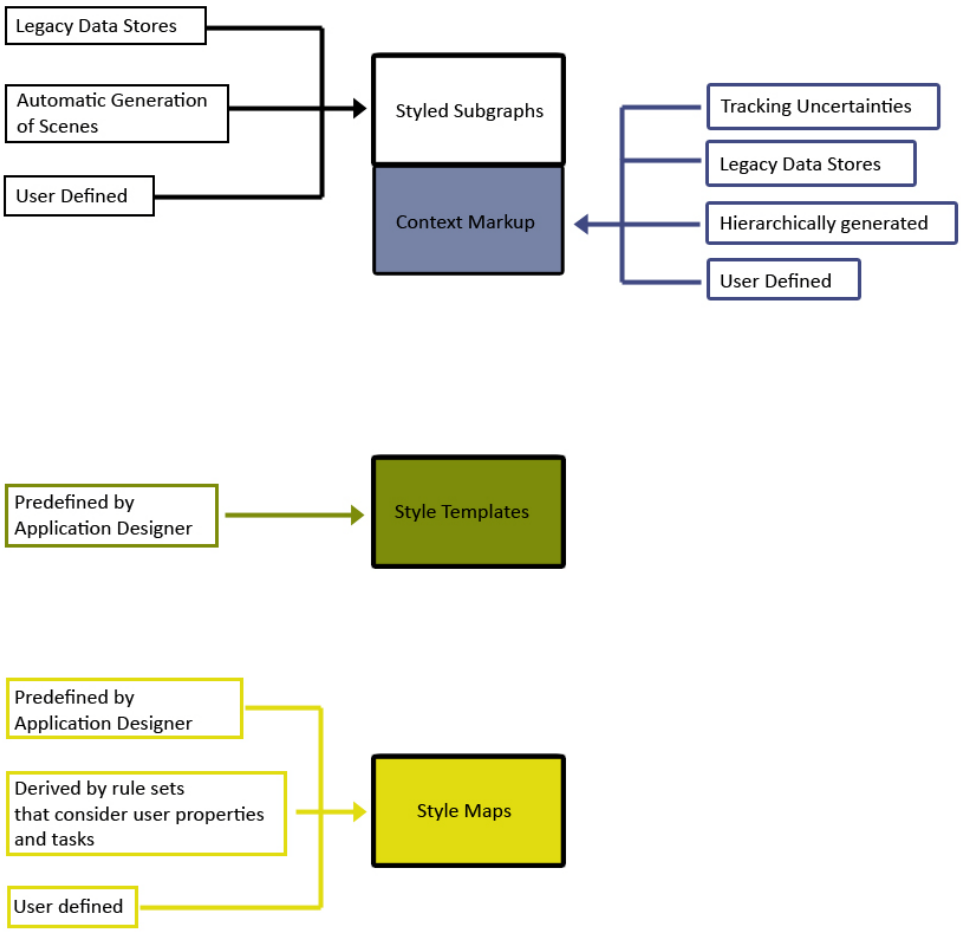


Figure 30. Example of possible sources of information for all four building blocks. These vary from user defined to high complexity rule sets.

3.3.2 Adaptive augmented reality scenarios

Consider an example application where the user has to repair a particular part of a car. The parts of this car have been enriched with contextual information, such as whether they are interior or exterior, left or right, seats or engine and so on. This application allows users to mark a family of objects for repairing based on their contextual attributes. For example, the user may be interested in repairing all the wheels, or just the wheels on the right side of the car. At the same time, dangerous objects are highlighted, so that the user can be made aware of potentially problematic situations.

This application defines templates for dangerous objects in red and for repair objects in green. Notice that the designer of the application has not specified which objects are, for example, dangerous. This is because such semantic interpretation of dangerous objects may not be known until the moment of execution of the application. The following code creates such templates:

```
StyleTemplates
{
 names  ["Dangerous", "Repair"]
 styles [USE RED, USE GREEN]
}
```

The mapping of styles and specific context families happens during the execution of the application. These mappings require a combination of context attributes that define a context family, and a style name to which this context family will be mapped. The following snippets achieve two different mappings:

```
StyleMap {
 templateName "Dangerous"
 keys ["Temperature", "Pressure"]
 values ["Hot", "High"]
}


StyleMap {
 templateName "Repair"
 keys ["Type"]
 values ["Wheel"]
}
```

Notice that the mappings of the “Repair” style to “Wheel” objects in this example is given explicitly. However, as mentioned before, this mapping may be also achieved by rule engines that consider user tasks, for example.

The last two members of our framework are the StyledSubgraph and the ContextAttribute. The purpose of the ContextAttribute is to define a set of key-value pairs at the moment of its traversal. The StyledSubgraph, in turn, defines the actual content to be traversed. The content itself may be a set of nested StyledSubgraphs with their own contextual attributes. The following snippet creates a subgraph for a wheel that was last changed in January:

```
Separator {
  ContextAttribute {key "Type" value "Wheel"}
  ContextAttribute {key "Changed" value "Jan"}
  StyledSubgraph {
    content
    {
      # actual content goes here
      # may be a nested StyledSubgraph
    }
  }
}
```

Similar to the style mappings, the values of the contextual attributes in this example are also explicitly given, but this may also be achieved by higher rule sets that consider external information or by specific combination of contextual values.

3.3.2.1 Application scenario

Based on the example outlined before, we now describe a more complex application scenario. This scenario includes three rendering style templates: dangerous, repair and neighbors. These three available styles change the appearance of the objects in the scene based on mappings and context markup that may be given by the user implicitly or explicitly, or defined a priori.

The templates in this case modify only the color and transparency attributes of those objects that fall inside a magic lens. Dangerous objects are colored in red when inside a lens and in half-transparent red when outside. This allows the user to always be aware of dangerous objects, regardless of whether they are in her work area or not. Objects for repair are, in turn, colored as green when inside the lens and half-transparent green when outside. Additionally, extra spatial information, such as the car's body, is shown in half-transparent white. This provides depth cues for interior pieces.

The mapping of contextual markup and templates is done via different mechanisms. For example, for dangerous objects a number of predefined mappings are available. Figure 31 shows a user selection of one of the mappings with a combo box (top). These mappings depend on temperature and pressure values of the styled subgraphs. The values are not predefined by the programmer, but are selected by the user at runtime.

In the case of repair objects, the mappings are not predefined, but the user is allowed to dynamically generate them given a selection of combo boxes. Figure 31 and Figure 32 show a sequence of steps that illustrate the dynamic bindings of styles. Figure 31 (bottom) shows that the user is interested in repairing the right front wheel of the car. Figure 32 shows the mapping changed to the left front wheel. Objects in the scene, whose context markup is being referenced, change their rendering style. In this case, the mapping has been changed from the right to the left wheel. At this point in time, the left wheel is shown as half-transparent green, since it does not fall inside the lens (Figure 32 top). Once the lens has been panned (Figure 32 bottom) and the wheel falls inside it, then the appropriate rendering styles are used. Notice that throughout the whole sequence, a half transparent outline of the car body is shown. This outline is, however, different from that of Figure 31 (top) that involves a different part of the car—the rear. This is because the last template ("neighbors") is

being mapped implicitly by the user, given her position in relation to the car. By doing this, we filter distracting artifacts that may needlessly increase the depth complexity of the scene.

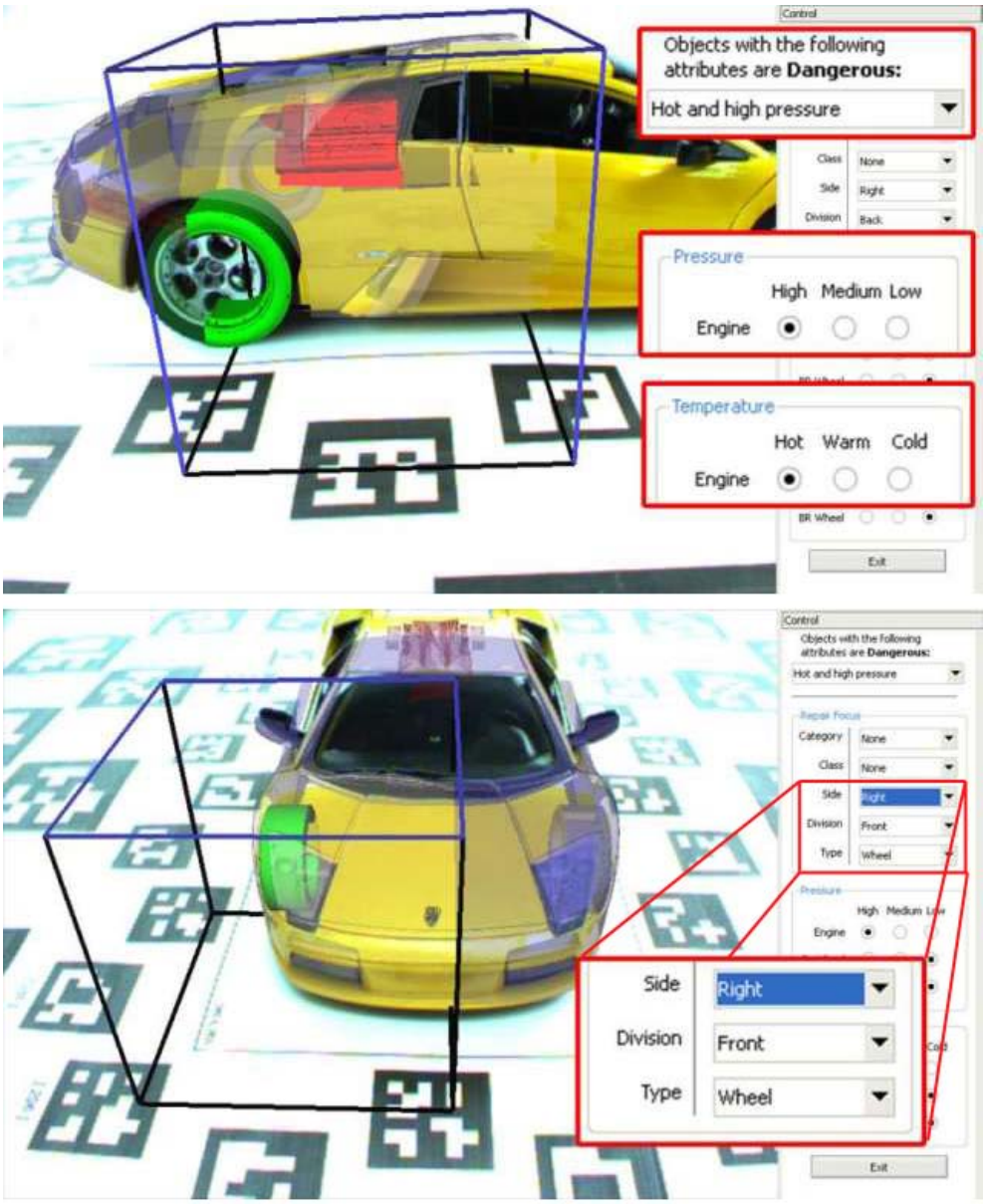


Figure 31. On the fly mapping. (Top) The user selects one of the predefined mappings and effectively changes the families of objects in the scene by modifying their context attributes. (Bottom) The user defines the rendering styles of objects by creating a customized mapping. Here, the right front wheel of the car is of the most interest. The magic lens is the black cubic wireframe.

In this application, the three style templates provide different visual information to the user. Some provide visual depth cueing, others highlight problematic pieces, and others draw attention to focus objects. This means that three tasks are running concurrently: spatial context, danger awareness, and priority targets. These three tasks are highly reminiscent of those outlined by Julier et al. (Julier et al. 2002).

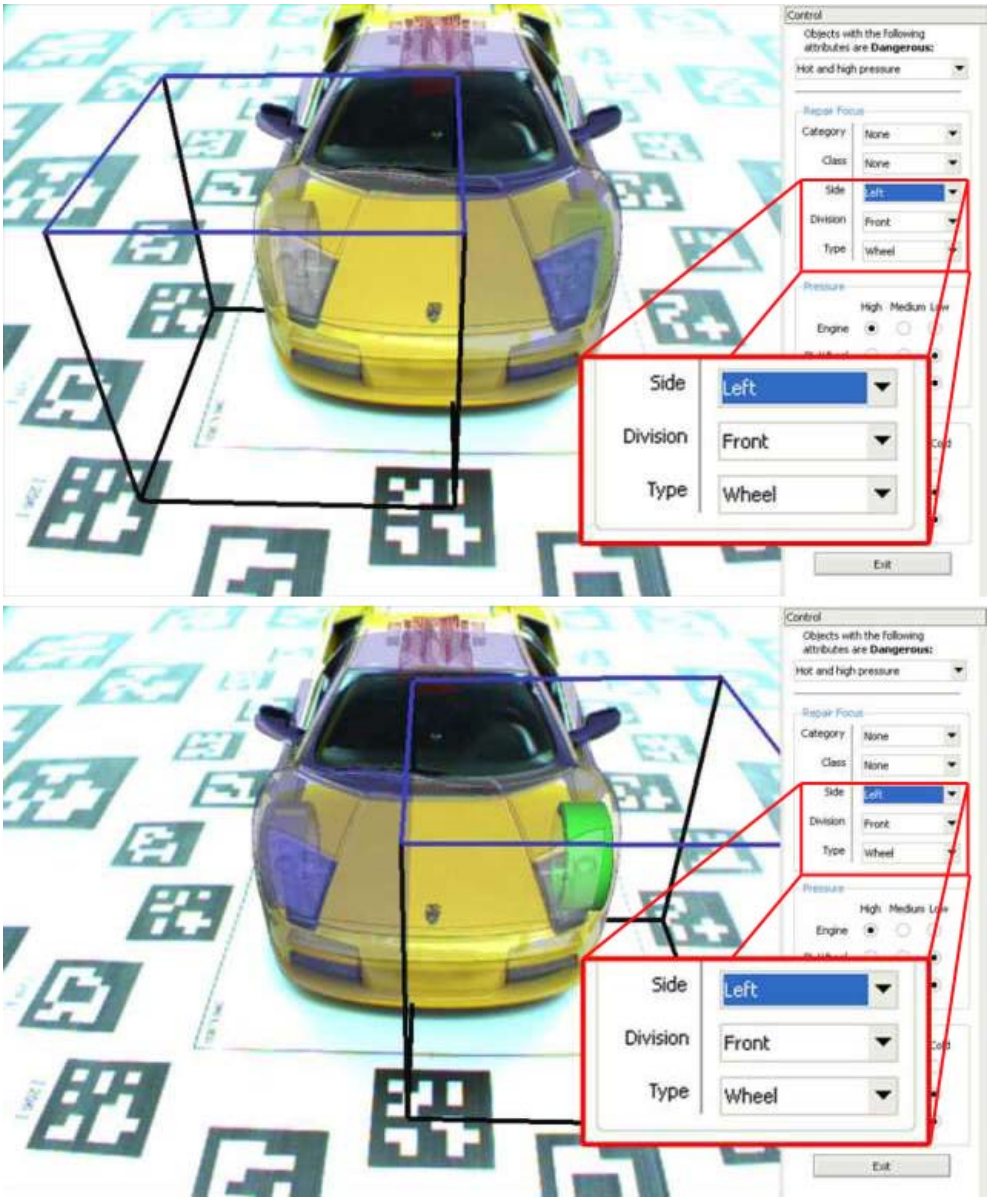


Figure 32. (Top) The user has modified the mapping to the left front wheel of the car. Dynamically, all objects in the scene fetch the appropriate rendering styles. (Bottom) A panning of the magic lens reveals that the appropriate rendering style is used.

3.3.2.2 Expected error as context

An interesting value to use as contextual markup is that of the expected error of fiducial tracking systems. A number of tests have been carried out to detect the accuracy of fiducial tracking in position and orientation (Malzebin et al. 2002; Abawi, Bienwald, and Dorner 2004). A simple error function presented by Abawi et al., depends on the distance and angle to the marker. For the sake of illustration, we will use this function at a constant distance of 30 cm. For this distance, the expected tracking error is:

- High error between 90° and 85°
- Low error between 85° and 20°

- High error between 20° and 0°

This angle is the orientation vector of the marker relative to the camera as returned by the fiducial tracking system and the normal of the marker, as illustrated in Figure 33.

Separation into two distinct families is a trivial task in this case. Objects whose position is being tracked by a marker will also set a contextual value called “Error” to the value returned by the previous function. In other words, objects will belong to one of two families: “Error=High” or “Error=Low”, depending on the angle to the marker’s normal and the values listed above.

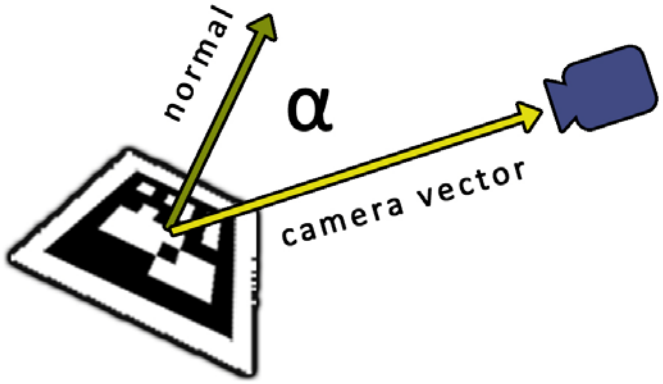


Figure 33. A simple error function for fiducial tracking checks on the angle between the camera and the normal of the marker.

We create two templates: “Reliable” and “Unreliable”. The first is set to a green color and the second to half-transparent red. The mappings are straightforward: Objects with a high error should be mapped to the “Unreliable” template and objects with a low error to the “Reliable” template.

Figure 34 shows three objects tracked by three different fiducial markers. Because of the angle between the camera and the normal of the markers, all of the objects have a low expected error. Figure 34, however, shows that two objects have a high expected error as defined by Abawi’s function (Abawi, Bienwald, and Dorner 2004). Notice that the grouping does not give any guarantee about tracking quality. This grouping is based only on a measure of the expected error. Although this is a simple example, contextual markup may be used by more robust uncertainty and error models.

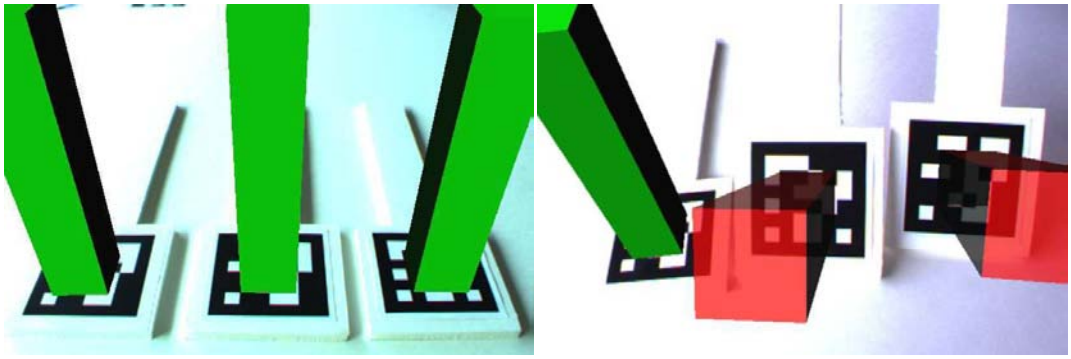


Figure 34. Expected error as context. (Left) Three markers with a low expected error. The error is calculated from the angle between the vector from the camera to the marker and the marker's normal. (Right) Two of the markers have a high expected error. This expected error is used as a context attribute, effectively creating context markup families on the fly, given tracking information.

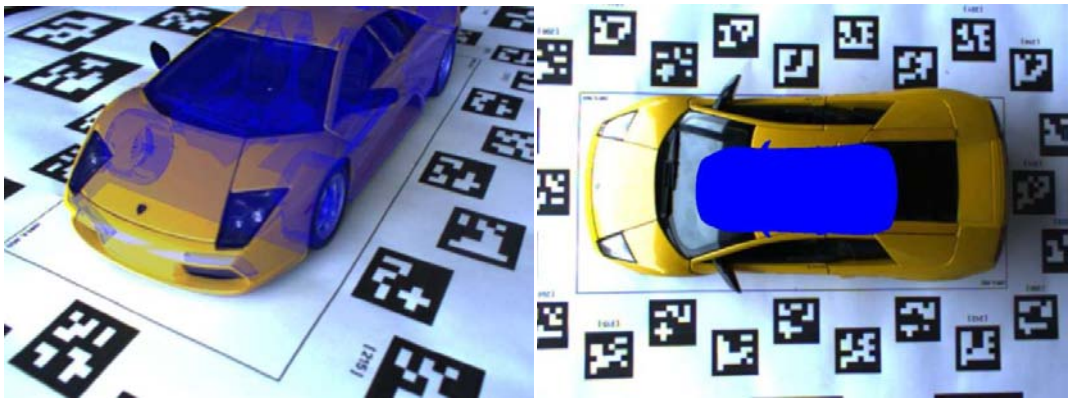


Figure 35. Expected error based real estate. Example application that shows how the combination of contextual markup derived from tracking uncertainties and rendering styles may aid annotation. (Left) The object has high quality tracking. (Right) An example of bad tracking. Objects with context markup defined by the accuracy of the tracking apply a scaling rendering style that reduces their "screen real estate". This real estate (in blue) may be used by annotations for correct placement.

A potential application of such context markup is to apply, for example, a scaling factor to the object. The resulting rendering of the object after such scaling may be used to identify "screen real estate" regions of the object similar to those used by Bell et al. (Bell, Feiner, and Hoellerer 2001) for annotation. Having a real estate region defined not only as a function of the object's shape, but of a mixture of its shape, tracking uncertainty and contextual markup, may lead to more powerful annotation techniques. Figure 35 shows an example application of this idea. Objects in this example create "screen real estate" regions that are used by annotations for correct placement. On the left, objects are not scaled because they have a low expected tracking error. On the right, we simulate bad tracking (exemplified by the orthogonal viewing angle to the markers). The rendered object in the scene is mapped to a down scaling rendering style that guarantees a real state region useful for labeling (in blue).

3.3.3 Discussion

Now that all the building blocks have been presented, it is important to underline the difference between traditional, context-sensitive (Section 3.2), and markup/mapping magic lenses. To exemplify these differences, we use an AR setup based on data extracted from a GIS database. This data includes building outlines, electricity and gas lines. Additionally we have placed one extra landmark for aided complexity. A magic lens has been placed in the scene and, to avoid image clutter, we do not display objects that fall outside the lens.

Bier et al. allowed styles to be assigned to objects depending whether they were inside or outside the magic lens. An example of this type of lens renders all objects inside it, for example, in red, whether they are of interest to the user or not. It will also render these objects regardless of their data structure arrangement or tagged contextual information. Although these kinds of tools are easy to develop, they provide little help in the task of information filtering. For example, it would be impossible to visually isolated objects of interest without prior knowledge of the scenegraph data structure.

In Section 3.2, we showed how objects in a scene enriched with contextual information can be affected differently given the same magic lens. That work showed how objects can be grouped by context, as well as hierarchically. That work, however, requires a prior definition of the available styles and does not separate these from the mappings with contextual families. Furthermore, membership in a contextual family is given explicitly by a single context attribute without hierarchical characteristics.

The technique proposed in this section highly extends the original CSML in two major ways: It allows a clear separation of style definitions and style mappings, and it allows contextual attributes to be hierarchically inherited. The two approaches mentioned before would not be capable of dynamically binding rendering styles to objects, given properties such as user interest or tasks.

Context-aware computing is popular in the ubiquitous computing (ubicomp) research community for creating adaptive user interfaces, but not so much in the AR community. Through the simple and well known mechanism of using markup to add context to scenegraphs, we hope to encourage researchers and developers relying on existing scenegraph solutions to adopt our approach. Like ubicomp, AR relies on sensor networks that can also be a source of context information.

Markup can be retrofitted to existing scenegraphs. While our implementation is based on Open Inventor, there is nothing in our work that precludes the quick integration of context markup in other scenegraph toolkits. It is largely compatible not only with standard scene-

graphs, but also with the modeling and interaction techniques typically used in conjunction with scenegraph–based engines.

The fact that application designers let users influence the styles of objects implicitly, instead of explicitly, will reduce the amount of interaction required. This may translate into simpler user interfaces, such as hands-free interfaces that play a prominent role in AR.

The separation of styled subgraphs from style templates with explicit mapping allows developing scenegraphs independently of style design and visualization policies. Style maps share the capability of XHTML cascading style sheets that a graphics designer can determine visualization styles once. These styles can then be applied to a variety of application and arbitrary content. This makes it easier to design complex content—a requirement not unique to AR, but very relevant if one wants to incorporate legacy data sources such as from the GIS with which we are currently working.

The next section outlines a more tightly coupled application scenario between the sources of content and the mappings between style and contextual information. It is strongly based on experiences working with geospatial data and provides more practical examples of the usage of a context-sensitive scenegraph.

3.4 Context influencing the procedural generation of models

We present now the final technique that uses contextual information combined with scenegraph styling. The work presented here, however, operates at a lower level by influencing the generation of geometric primitives. This approach uses context-sensitive scenegraph traversal combined with a generative model engine (Havemann 2005). Most of the work in this subsection is demonstrated experimentally with a well populated database of geospatial data. In particular, this database contains sub-surface infrastructure, such as water pipes and electricity lines. The data was obtained from utility companies and the examples given also refer to typical utility scenarios, such as the maintenance of infrastructure.

3.4.1 Design considerations

Large geospatial databases are the result of hundreds of person years of surveying effort, and rendering engines constitute highly advanced and optimized software toolkits. What is needed to connect both is commonly called *transcoding* in the GIS community: the process of turning raw geospatial data that is mostly two-dimensional with abstract non-visual attributes, into three-dimensional models suitable for standard rendering engines. Thus, the required transcoding is not simply a one-to-one conversion from one format to another. The work presented here requires that transcoding preserves contextual information throughout

the pipeline and three-dimensional models of pipes retain information about property, ownership and so on.

The overall objective of a three-dimensional model for visualization of geospatial data is to provide comprehensible visualizations of the assets in question. Since there is a large variety of geospatial objects and possible visualization styles, we desire a system architecture that allows content types and visualization styles to be added as plug-ins of the actual client. Ideally, a compatible three-dimensional browser should be capable of loading and displaying self-contained content. Moreover, the browser should be capable of displaying user interfaces for the selection and manipulation of parameters influencing the application. While these requirements apply to many visualization applications with complex datasets, this section specifically aims at addressing the following requirements:

- *Appealing shape.* For example, intersections of branching pipes must be continuous. This cannot be achieved by simply converting raw vectors from the geospatial database into cylindrical tubes (i.e., more advanced modeling methods are required).
- *Flexible styling.* A user may choose styles for individually selected objects or groups of objects. Styles are also dependent on the viewing situation: AR displays may require changing the styling parameters depending on the quality of video see-through and registration. To allow flexible handling of styles, the styles should be stored with the content and not be a feature of the specific three-dimensional browser. Clean separation of styles from content makes styles reusable across multiple types of content (see Section 3.4).
- *Progressive information revealing.* Semantic level of detail is a technique that can manage screen real estate efficiently by using multiple representations of the same object that progressively reveal more visual and functional detail. For example, objects can be arranged in containment hierarchies, such as in the case of underground infrastructure where cables are arranged inside casings that, in turn, are contained in tubes, and so forth.

A straightforward conversion of the geospatial data into a static polygonal representation can address only some of these requirements. And even a dynamic multiresolution tessellation will not help in addressing the needs for interactive manipulation and control. Such interactive capabilities can be added as a custom feature to the three-dimensional browser, but this approach makes content and browser highly interdependent and defeats easy extensibility.

Therefore, we have opted for a combination of techniques that are implemented as extensions to a conventional scenegraph library. New object types are handled through built-in

interpretations of the scenegraph structure, and do not require any modifications of the scenegraph browser itself.

1. Scenegraph level: Scenegraph markup (Section 3.1) enables us to attach the semantic attributes from the geospatial database. The markup also provides the hooks for interactive high level control provided by the user, and allows filtering and styling, as well as a certain degree of semantic Level of Detail (LOD) control.
2. Object level: To provide high quality tessellated objects with geometric and semantic LOD control, we added a new type of scenegraph node with a lightweight embedded interpreter for a stack-based language, the generative modeling language (GML) (Havemann 2005).

Semantic attributes are passed as parameters from the scenegraph to the GML nodes during scenegraph traversal, as described in Section 3.3. In the remainder of this section, we present the components of our information pipeline: the transcoding from the GIS data to our scenegraph format, the GML, and the scenegraph markup that links scenegraph and GML together.

3.4.1.1 Transcoding

The first step in our information pipeline is a transcoding pass. This means a change of the data format, in our case from a GIS encoding to a scenegraph enriched with GML nodes. The result of the transcoding is a scenegraph description with per-feature grouping of shape objects and semantic attributes. The shape objects refer to embedded GML scripts for dynamic parametric objects, and to scenegraph classes for static non-procedural shapes. The transcoding pipeline focuses on common features found in the underground infrastructure.

This dissertation does not focus on the transcoding process itself, a complex technique that involves querying and parsing GIS information. Instead, it assumes that the transcoding has been successful and that the resulting scenegraph enriched with contextual information and GML nodes are the input for the next steps.

3.4.1.2 Scenegraph markup with semantic attributes

The result of the transcoding step is a scenegraph with semantic attributes. This has an important advantage: traditionally, in order to highlight all objects of a certain type inside a scenegraph (e.g., by changing their color to red), the respective material properties of all affected nodes must be changed somehow. This can be done by changing a material field of the node, or by inserting material nodes into the scenegraph. However, a much better strategy is to change only one “style node” for all the desired nodes. The changed parameters are propagated automatically when the scenegraph is traversed for rendering: Each affected node

updates its styling because its attributes have been touched. This is described in detail in Sections 3.3.1.1 and 3.3.1.2.

Objects in our scenegraph are not only grouped by the hierarchy of the graph, but also by families defined by their semantic attributes. Objects from the same family do not have to belong to the same part of the scenegraph hierarchy—they can be scattered throughout the scenegraph and still be jointly affected (Figure 36). Additionally, the family membership of an object is not explicitly given, but is the result of an aggregation of semantic attributes encountered while traversing the path from the scenegraph's root node to the object itself (see Section 3.3).

This mechanism works exactly like other traversal states such as matrix transformation or material bindings. Semantic attribute nodes encountered during traversal add or overwrite arbitrary key-value pairs in the current attribute set. This set is propagated downwards with traversal, and can be queried by any node aiming to be contextually sensitive.

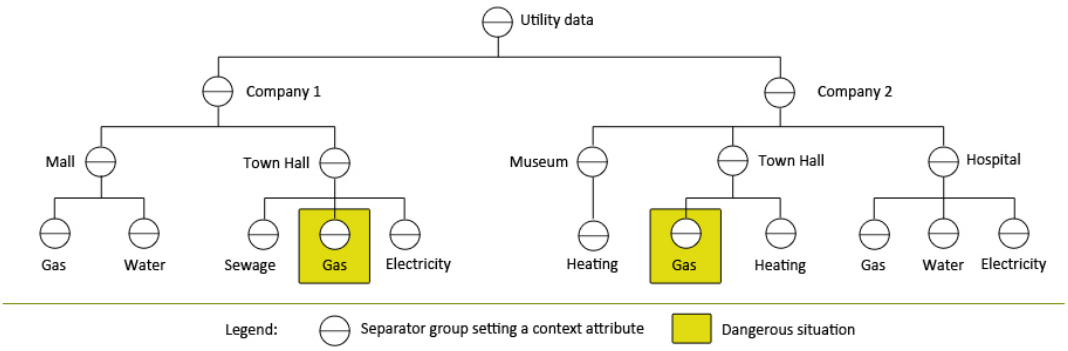


Figure 36. A scenegraph with semantic attribute markups (similar to the graphical style sheets of Beach and Stone (Beach and Stone 1983)). The Town Hall's reconstruction requires welding that must not occur near Gas. By combining the attributes (Town Hall + Gas), the system automatically locates and highlights dangerous objects during a single scenegraph traversal.

More specifically, a styling node relies on a style map to modify appearance parameters for the following shape nodes. The style maps are also hierarchically organized in a way similar to Cascading Style Sheets (CSS) used in XHTML. Entries in the style map are "style" subgraphs consisting of nodes influencing appearance that are dynamically inserted before the object to be styled during scenegraph traversal. In that way, object appearance changes dynamically with the associated semantic attributes. The advantages are:

- *Preserving semantics.* Semantic attributes from the geospatial database are retained until the actual rendering traversal, and used in this stage to determine appearance. Existing styles can be applied to new types of objects if they use compatible attributes.

- *Referencing.* Potentially complex combinations of semantic attributes can influence the appearance. Consider a situation where construction requires welding that must not happen close to a gas pipe for security reasons. A style demarcating "danger" can be automatically selected if attributes "gas" and "welding" are applied to an object group in spatial proximity.
- *Interactive styling.* Selected semantic attributes can be exposed in the user interface, since they are just key-value strings. The user can directly modify attribute values and thereby influence the visualization. For example, filtering of distracting objects can be achieved by setting the style for the distracting object category to "off."
- *Parameter forwarding.* The semantic attributes can be forwarded as parameters to the GML engine creating the geometric primitives. More details are given in the following subsection.

Geographical features are encoded as small subgraphs inside the scenegraph containing both its semantic and geometrical attributes (Figure 37). Semantic attributes add to, set, and modify key-value pairs in the traversal state. This provides to any geometrical attribute the ability to check whether any mapping exists that matches its semantics. Every geometrical attribute is in itself a subgraph containing a styling and a content node. If a mapping exists, it modifies its styling node in order to affect the appearance of the content node.

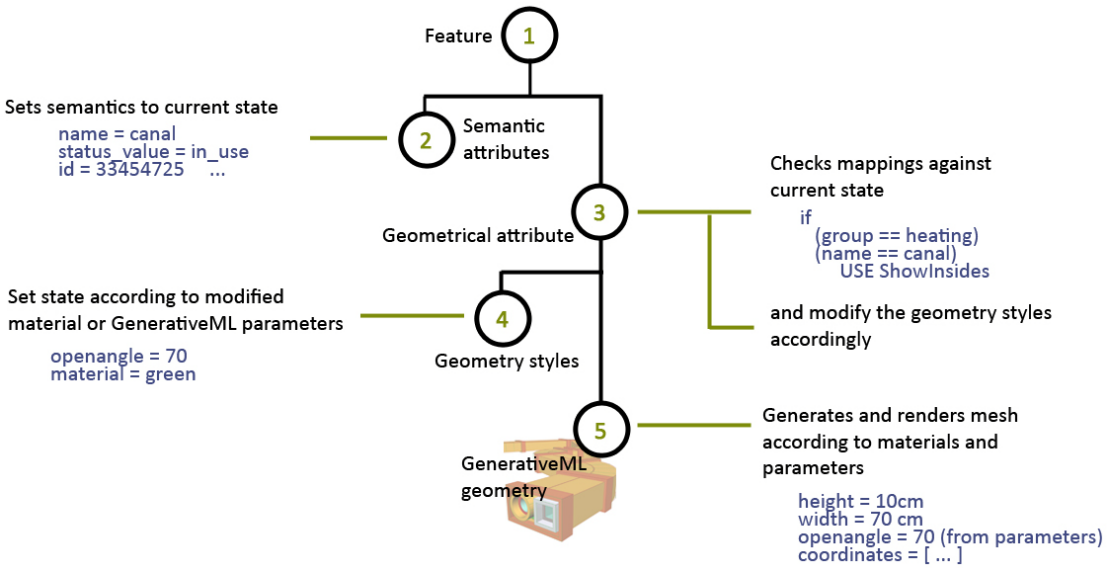


Figure 37. Traversing geographic features. Upon traversal, every geographical feature aggregates its semantic attributes to the traversal state, and checks whether the current total of the semantic values match a style mapping. If so, it fetches predefined styles accordingly. Conceptually, the shape node's rendering style reflects its semantic attributes. The numbers inside the circles reflect the traversal order.

3.4.1.3 The generative modeling language

In our system, the semantic attributes not only affect the appearance of three-dimensional objects, but also their construction parameters. The technical device we use is the Generative Modeling Language (GML), a simple stack-based scripting language for creating parametric three-dimensional models on the fly (Havemann 2005). Its original purpose is to serve as a general exchange format for procedural models (e.g., as a file format for encoding the construction history of complex objects). It is capable of generating large amounts of geometric data out of compact descriptions. Since the syntax is similar to the Adobe PostScript language, GML can be thought of as a kind of "3D PostScript."

The GML interpreter and its integrated OpenGL renderer are contained in a shared library that can be linked with any OpenGL application (e.g., a scenegraph engine). In this way, GML has been embedded into our scenegraph as a new node communicating with the GML engine. Script code is stored and maintained in a string field of the GML scenegraph node in the scenegraph file. This code can be executed during traversal to produce three-dimensional models of pipes and tubes on the fly. This greatly reduces scenegraph size and makes it possible to vary construction parameters to reflect changes on the semantic level, such as highlighting or database queries.

Note that generative modeling is complementary to scenegraphs with semantic attributes: The strength of a scenegraph is the management of large scenes, providing tools such as scene hierarchy, spatial structuring and flexible scene traversal. In contrast, GML operates on the object level. Its purpose is to create massive amounts of geometry based on rules and parameters. The interface between both software components is just the set of semantic attributes that are passed as model parameters to GML.

3.4.2 Application to underground infrastructure

Consider the following task from the public utility sector. Utility companies need to provide private contractors, such as construction companies, with spatial information about their assets. This is essential to avoid damaging existing infrastructure during digging. Therefore, a common duty for field workers employed by utility companies consists of spray painting spots on the ground where digging should take place. Orientation at the construction site is performed with paper maps plotted in advance using the utility company's GIS.

3.4.2.1 Adaptive modeling of assets

The power of GML is that it supports processing chains quite efficiently. The stack is a flexible way of passing data produced by one function as input parameters to the next. This is exploited to create different types of pipes simply by passing different profiles to a connecting function.

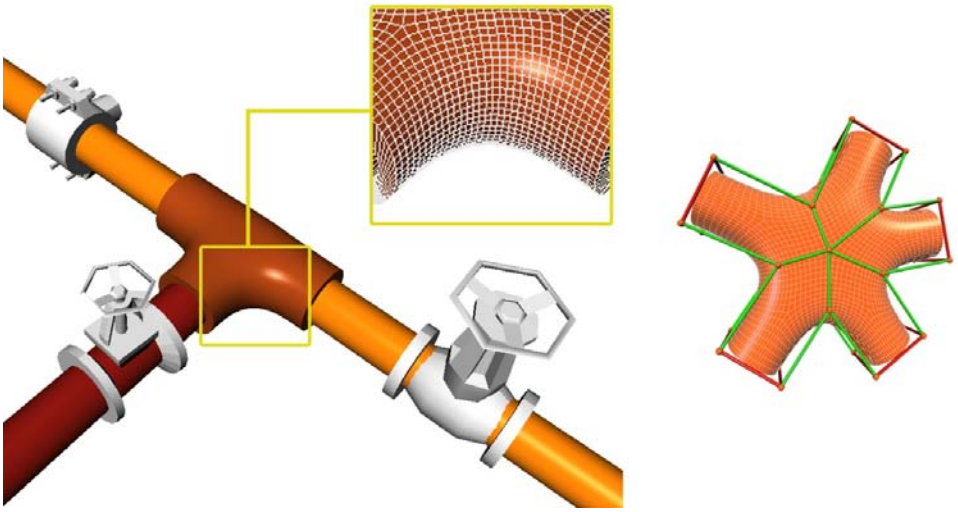


Figure 38. Combining modeling techniques. This image is a composite model of a GML T-junction with valve models from a stock library. (Right) By using subdivision surfaces, GML can create complex junction shapes such as the inset's five-way junction.

The interactive LOD enables us to generate more visually pleasing images. For example, intersections of pipe junctions are given as sharp connections in the geospatial data. GML models can soften these junctions by further subdividing the resulting mesh. Figure 38 shows an example of a combination of the different modeling techniques. The pipes and the T-junction are modeled using GML. (The smooth joint makes the connection clearly visible.) The white objects are predefined models from a shape library that are low resolution to avoid impacting the rendering performance. The geometrical positions of all these models, and their orientation and semantic attributes, are also extracted from the geospatial database by transcoding.

3.4.2.2 AR-style visualization

Next, we will sketch a practical solution for the public utility sector. We assume that the field worker can perform online queries while working on the construction site by using a wireless mobile computer that sends her current GPS position. The server then returns a set of geographical features. Currently, field workers employ handheld computers that are capable of displaying 2D map information more or less equivalent to conventional paper maps.

The positions of pipes, ducts, valves, alignments and other underground infrastructure are exclusively based on data from the utility company GIS, since they cannot be acquired using aerial or terrestrial photography. Above surface information, such as building footprints, roads or property lines is also taken directly from the geospatial database. We mainly consider the delivery of the final visualization as part of a video-see-through AR display, but the visualization techniques are also applicable in desktop virtual reality applications in a planning office.

Figure 39 and Figure 40 show geospatial data superimposed on a video feed using the AR interface. The visualized information represents heating assets, organized in tunnels and canals containing pipes. A simple task is to request all the objects of type canal to show their interiors. This action sets the GML parameter "opening angle" to 70° only on those objects with matching semantic attributes (i.e., canals). Notice that not all the objects have opening angles—those that have, are set to green (Figure 39 bottom). This action has revealed the pipes inside the green conduit that is cut open. Alternatively, we may change the transparency of objects so that they show their interiors as illustrated (Figure 40).

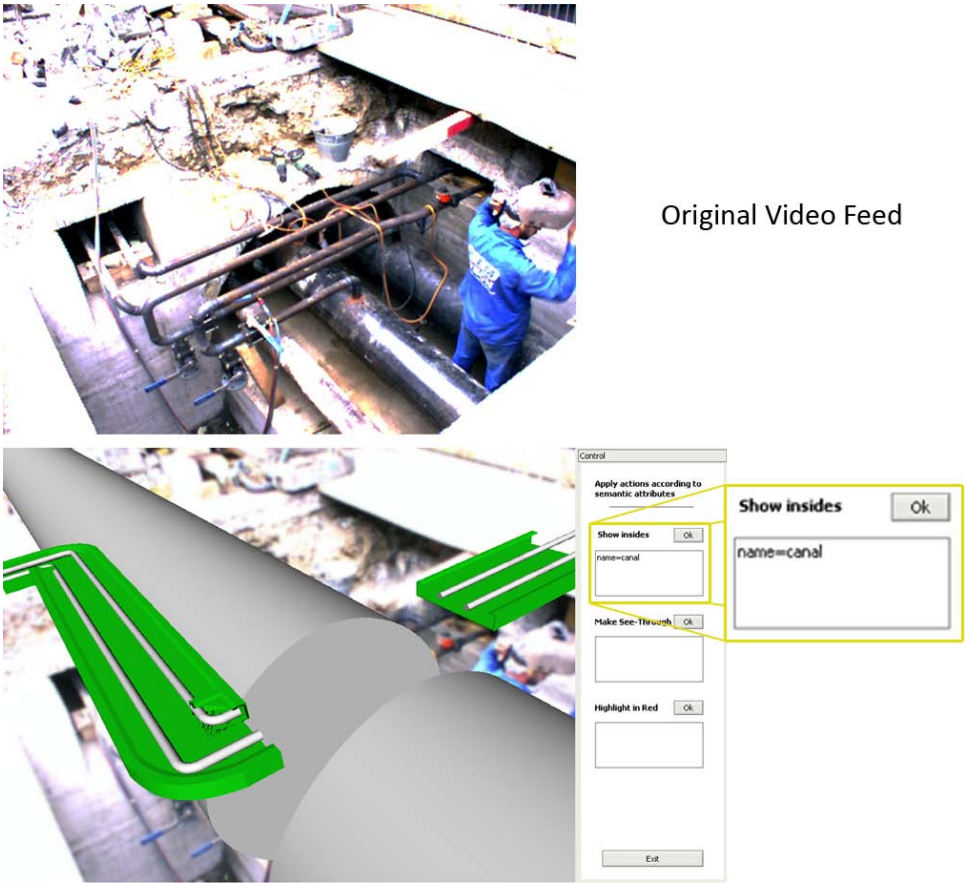


Figure 39. Visualizing the superimposed geospatial underground model. (Top) In this photo of a construction site, the district heating pipe infrastructure's supply and return pipes are clearly visible. The technician's task is to repair some of these pipes. (Bottom) The technician requests that all objects of type canal (in green) show their interiors.

Field workers usually need to refine their search using additional attributes, such as objects that are currently in use, or that belong only to a certain category. In the dialog of the top image of Figure 40, the user would like to open only those tunnels used for heating and that are in use. Different styles may be assigned to objects. Such styles are created by an application designer and may convey subjective meanings, such as "dangerous" or "important." Fig-

ure 40 (bottom) shows the user instructing the system to highlight an object of a specific ID that, in this case, is targeted for repair.

The benefits of this visualization are twofold. First, underground infrastructure can be located and visualized before an excavation is performed—although in this particular example, for the sake of illustration it is done after. This allows visualizing buried assets, for example, to locate damages or during construction planning.

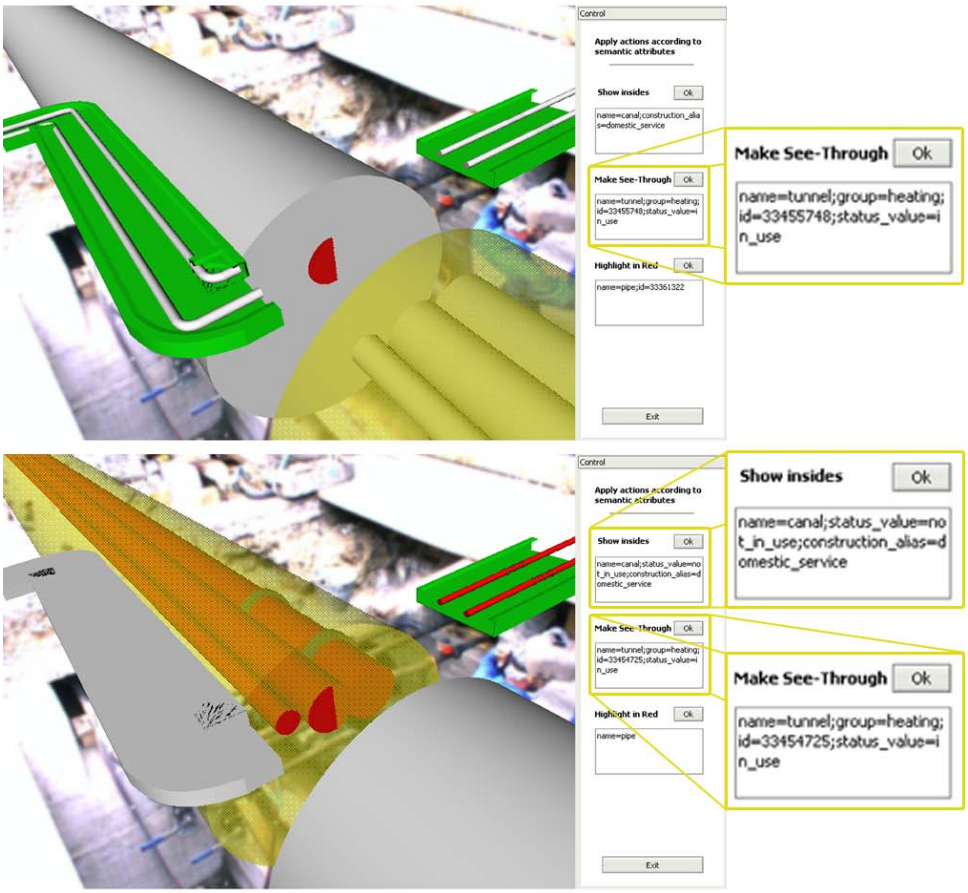


Figure 40. Visualizing the superimposed geospatial underground model (continued). (Top) The technician tells the system to show the insides of the canal and the tunnel used for heating. (Bottom) The technician asks the system to highlight the object with a specific ID.

Second, visual guidance during excavation activities and of technical infrastructure can be achieved. Information about properties of the pipes can be retrieved on demand (e.g., purpose, specification, length and material of pipes, as well as construction date, and companies involved in maintenance). A technician using this system would no longer need to contact the company's back office or consult printed documentation when information related to the pipes or to the construction site is required. Due to our data transcoding technique, all properties of the pipes that are stored in the corresponding source geospatial database are avail-

able for information visualization. Consequently, the on-site situation can be assisted more intuitively.

3.4.3 Discussion

One problem with the design presented here is that the information flow is strictly one way from the scenegraph to the GML nodes that generate the geometry procedurally. The scenegraph itself is static in the sense that, at runtime, no nodes are added or deleted; only the connections between these nodes (or subgraphs) can be changed at runtime. More flexibility can be obtained if it is also possible to create parts of the scenegraph procedurally.

A limitation on the technical level is that our current scenegraph engine renders each object immediately when it encounters it during traversal. Since a pipe is composed of several different materials, that implies a great number of material state changes. This does not allow rendering optimizations such as collecting from different objects all faces with the same material. The semantic markup allows the styling of objects during the traversal itself without prior knowledge of their position in the graph. This can cause caching problems, since every subgraph has to be reformatted to reflect its semantic style mappings upon traversal.

This section presented a system for extracting geospatial data of underground infrastructure and for creating interactive models for AR displays. We believe that the approach of retaining semantic information and using it not only for textual annotation, but also for influencing three-dimensional shapes and visualization styles, has wide applicability in modeling of human-made structures from existing legacy data.

The decision as to the amount of information to be reformatted during transcoding implies a trade-off between performance and flexibility. Preserving semantic information down to the traversal stage increases by far the flexibility to apply visual and modeling changes to the objects in the scene. But it also implies a decrease in performance: The number of traversals increases, since every node is adapted to reflect its semantic mapping by traversing a styling node prior to the geometrical content. A better strategy is to consider specific user tasks (as in the case of infrastructure maintenance) that would reduce as much as possible the number of semantic attributes that need to be preserved by the transcoding pipeline.

3.5 Summary

This chapter presented a series of techniques that use contextual information to change the visual results of AR applications. The work is highly inspired by collaborations with utility companies. The first technique enabled the use of different visual styles of the same object depending on contextual information and a magic lens. Examples are given for information filtering, information revealing and an X-ray tool. The second technique provided a more

general definition of style mappings and templates that further enabled the decoupling of visual representations and the input information. Finally, the last technique made it possible to influence the generation of geometries during traversal, but before being sent to the rendering engine.

Much work remains to be done. For example, data transcoding was simply outlined, but no details are given in this chapter. The techniques presented provide a flexible basis for AR, but performance can be improved. Caching of data is difficult when retargeting subgraphs that are shuffled during traversal. State changes on the rendering engine are always expensive operations that need to be minimized; in the current described techniques this should be taken into account.

The current matching between styles and contextual information is done by matching one or more pairs of strings. This can, however, be improved to more advanced matching techniques that consider, say, the current traversal state. For example, objects that are currently being manipulated by the user could be rendered in a different style. Another example would be objects that are currently being loaded from file or downloaded could also be matched to a specific rendering style.

And finally, the biggest problem is the lack of standardization of data descriptions. GIS data bases are populated by multiple companies with different documenting strategies, and, even within a company, different people are in charge of data surveying. This results in problems from naming conventions to entire data structure arrangements. This is an inherent problem in the GIS community that application designers must face.

Chapter 4

Using Context to Enhance the Perception of Depth

The previous chapter introduced the notion that contextual information should be preserved to allow better targeted rendering styles late in the traversal. Although the definition of context used here is slightly different, this chapter suggests again that contextual information should be preserved. The goal this time is not for rendering styling, but to enhance the perception of depth in the scene.

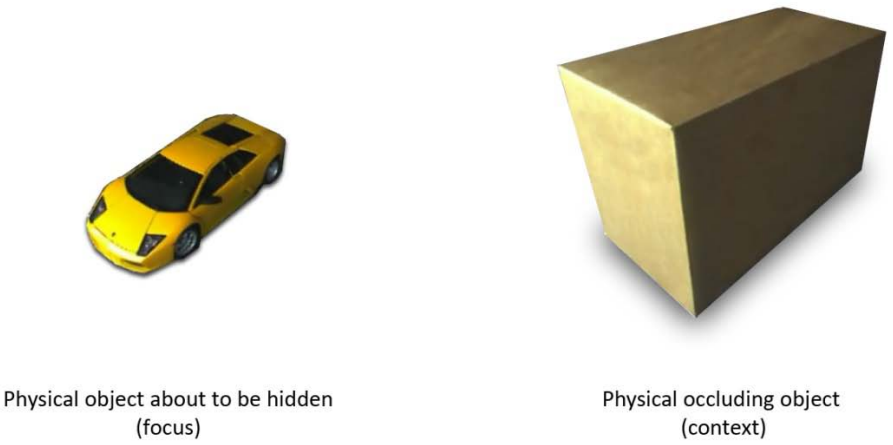


Figure 41. Physical objects. (Left) The car illustrates a physical object that is of interest to the user. (Right) The box is a physical object that can be placed on top of the car, thereby occluding it. This simple setup is used for illustration throughout this chapter.

There are two major differences with the previous chapter: the definition of what is context and the final purpose for preservation. In the previous chapter context was *abstract information* in the formed of tagged textual values on digital data. For example, a building's ownership and date of construction are referred as tagged contextual information. In contrast, in this chapter contextual information is *spatially occluding information*. The goal of the previous chapter was to exploit the existence of contextual information to modify rendering styles upon runtime; thus, *information filtering* was one of the main topics discussed. Instead, this chapter prevents the careless overwriting of occluding contextual information (also called *context preservation*) in order to improve the spatial perception of objects.

In AR, careless replacement of portions of the physical world image leads to a misrepresentation of the intended spatial arrangements. *Clutter* is the most common example of this situation, where an excessive amount of computer-generated information is added. This causes an overload of visual information that is counterproductive for a system. Another common problem happens when a poor occlusion-management technique is used. For example, AR allows us to reveal hidden physical structures by overlaying a digital representation in place. Take, for example, Figure 41, in which the left physical object is of interest to the user. This, however, may be hidden by another physical object, such as the box on the right. A rudimentary AR application would simply overlay a virtual representation of the physical object, such as illustrated in Figure 42. This can be problematic, as important information may be occluded by the new augmentations. Studies have shown (Interrante, Fuchs, and Pizer 1996) that merely overlaying a virtual representation of the object that is about to be occluded disrupts the perception of distance and spatial arrangements across the objects in the scene: this is causes a problem of *depth perception*.

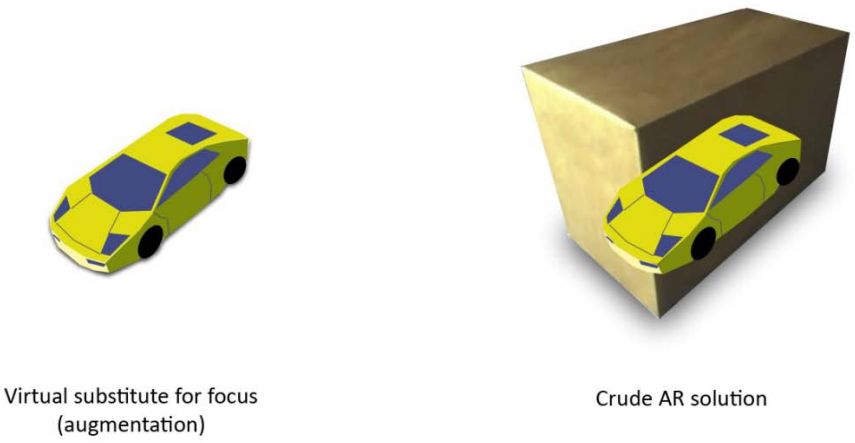


Figure 42. Virtual model. (Left) The virtual model will serve as a substitute augmentation for the hidden physical car of Figure 41. (Right) The illustration represents a crude AR solution in which the occluding object is merely displayed on the image.

The depth perception problem has been a common subject of research in the past and has been addressed by several strategies, typically using a monocular display (Goldstein 2006). For example, a window restriction is an effective and easy to implement and effective technique, showcased by Furmanski et al. for ameliorating depth perception issues (Furmanski, Azuma, and Daily 2002). The main idea of that technique is to partially restrict the visible portion of the augmentation to where the user can see it, thus exploiting motion parallax. Bichlmeier et al., for example, use a squared restriction that can be moved independently of the user's location (Bichlmeier and Navab 2006), while Viola et al. use an arbitrarily shaped

restriction attached to the user's field of view (Viola, Kanitsar, and Groeller 2005). Although window restrictions do a good job at communicating spatial arrangements, the problem remains that physical information is carelessly overlaid by digital information. Other techniques, such as explosions and cutaways, take a different direction by digitally moving the location of virtual or physical objects in the image (Li, Agrawala, and Salesin 2004). These techniques effectively communicate the interdependent spatial arrangements of virtual and associated physical objects, but introduce a new class of problems by moving objects from their physical anchor.

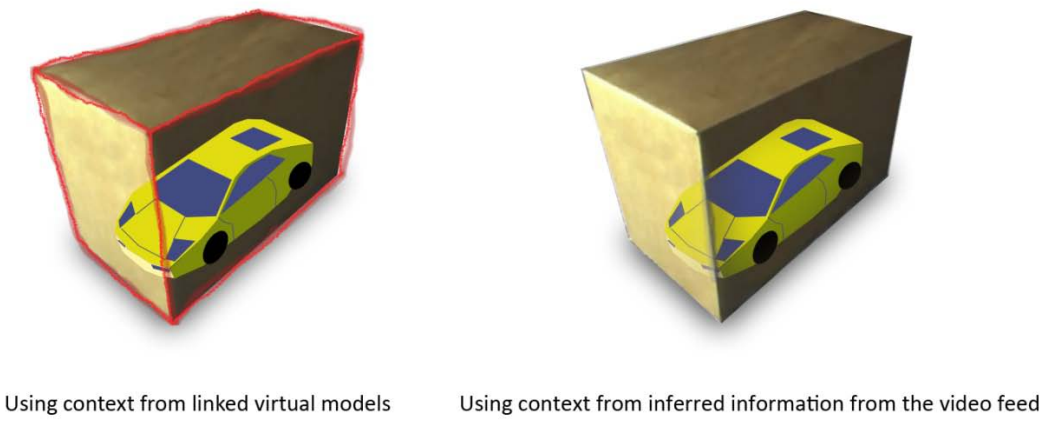


Figure 43. Illustrations of the solutions presented in Sections 4.3 and 4.4. (Left) A simplified representation of a solution based on linked virtual models. (Right) A simplified version of context preservation by inferring information from the video feed.

In this chapter, we address the problem of depth perception by taking into consideration information from the physical world that is to be occluded. The problem of showing physically hidden information with a digitally visible counterpart has been referred to as *information revealing*, *X-ray visualization*, *occlusion management* or *focus and context*. Focus and context techniques refers to the splitting of the scene into two categories, those of interest to the user (focus) and those that mainly provide a reference frame (context). A crude classification is that all hidden objects that we want to reveal are part of the focus, while the rest becomes part of the context. In this chapter, we will introduce the idea that such binary partitioning of the scene can be harmful to our visualizations. Instead, a multi-level focus and context description is given.

To address the problem of augmentations occluding useful real imagery, we introduce the notion of *context-preserving focus and context*. We propose the idea of carefully overlaying augmentations while considering the contextual part of the image. Instead of simply display-

ing synthetic data, we extract important information from the context and preserve it in the final image. The question is what information should be preserved and in what amount.

The work presented in this chapter tackles this problem with three different approaches: (1) by preserving context from virtual models linked to physical ones (Section 4.3), such as illustrated in Figure 43; (2) by preserving context from the video information (Section 4.4), as illustrated in Figure 43; and (3) by preserving context, given a predefined or procedurally generated mask (Section 4.5), illustrated in Figure 44. All three techniques work on the same principle, but use different methodologies for how to define the information that should be preserved and the quantity. The next section will describe a common concept used throughout the rest of the chapter, *two-level and multi-level focus and context*. The following sections will then build on top of this definition.

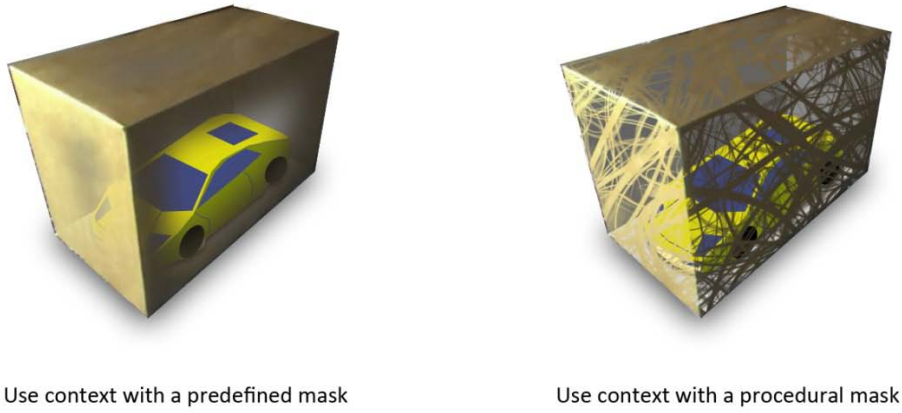


Figure 44. Illustrations of the solutions presented in Section 4.5. (Left) A simplified representation of a solution based on predefined masks. (Right) A simplified representation of a procedurally generated mask.

4.1 Two-level focus and context

Focus and context techniques typically demand a binary classification of the scene objects. Anything in the field of view must be either labeled as being of high interest to the user, referred to as *focus*, or regarded as merely a frame of reference, referred to as *context*. Figure 45 illustrates this classification, where scene objects are defined as either focus or context. The problem with this classification is that it considers scene objects as a whole, without a detailed ranking of parts within each category. Thus, when a book on a shelf is cataloged as focus, information such as the shelf's section tag is disregarded.

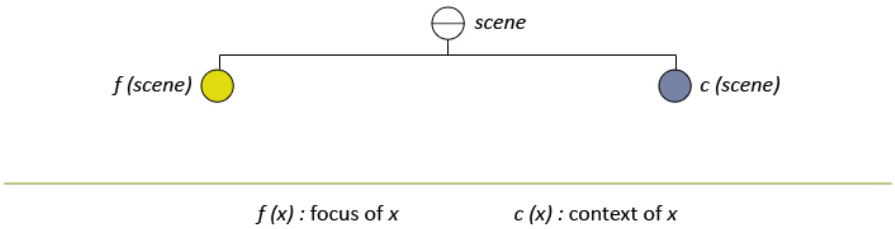


Figure 45. Binary focus and context classification. The scene is classified into two groups: the focus of the scene and the context.

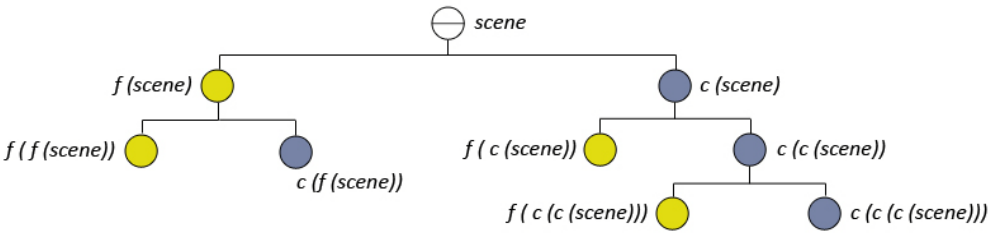


Figure 46. Multi-level focus and context. This image illustrates a more advanced classification scheme for focus and context, based on recursively dividing every node as either focus or context. Thus, *f(c(c(scene)))* is the focus of the context of the context of the scene.

What is needed is a hierarchical classification of focus and context; which we refer to as *multi-level focus and context*. This technique is conceptually similar to the concept of a *binary space partitioning tree*, where a virtual space is recursively subdivided into convex sets (Fuchs, Abram, and Grant 1983). In multi-level focus and context, we recursively provide a definition of what is focus and what is context; hence, within each object labeled as context, there are parts of it that are more important than the rest. This is a simple, yet effective definition that allows a visualization technique to determine whether it has to consider the information about to be occluded and in what amount it can be occluded. Figure 46 illustrates multi-level focus and context, in which every ramification of the scene is subsequently subdivided into focus and context parts. Notice that this graph is not a scenegraph representation, but simply an illustration of the classification of scene objects. In practice, objects in the same classification may be in separate portions of the scenegraph, or may not even exist until runtime. However, a recursive binary classification of scene objects is not the only possibility. The work by Julier et al. on focus and nimbus (based on the work of Benford et al. (Benford and Fahlen 1993)) suggests a softer boundary classification in which the focus has an "influence" range, or nimbus, in which other focus regions can affect it (Julier et al. 2002)". That classification could be complementary to the concept of multi-level focus and context.

The idea of multi-level focus and context was introduced in 2007 by Kalkofen et al. (Kalkofen, Mendez, and Schmalstieg 2007). It is a general concept, where every level in the hie-

rarchy should receive special treatment. That concept, however, may be reduced to two levels of focus and context. This two-level classification requires a definition of 1) the *focus of the scene*, 2) the *context of the scene*, 3) the least important elements of the context of the scene, or the *context of the context of the scene* 4) the most important elements of the context, otherwise referred to as the *focus of the context of the scene*. This last classification group will represent the objects within the context that should receive special treatment by the visualization technique. This simplified definition is sufficient to improve the perception of depth for single layer occlusions, but it is inadequate for dealing with multiple layer occlusions. Throughout this chapter, we will assume that there is only one layer of occlusion between the user and the focus object. Moreover, for simplicity's sake, we will assume in our examples that there is only one focus object (about to be hidden) and one context object (about to occlude). Figure 47 illustrates the two-level focus and context classification.

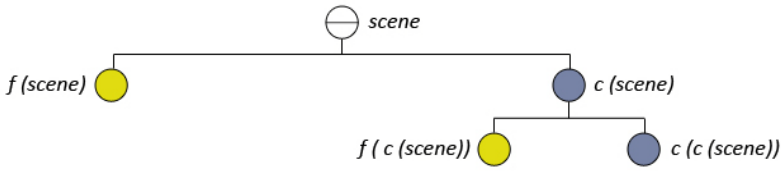


Figure 47. Two-level focus and context. This simplified form is sufficient for enhancing the perception of depth of single layer occlusions. In addition to the classical binary classification it also adds a definition for the most important parts within the context of the scene.

We require the definition of a third classification group for our focus and context techniques. Information in this classification should specify the portions of the context that are of high value, which we will occasionally refer to as *importance information*. Objects belonging to this classification can be of any type, three-dimensional models, volumetric data or merely two-dimensional masks. Section 4.3 uses an untextured three-dimensional object to define the importance information, while Section 4.4 uses a screen-aligned two-dimensional mask, and Section 4.5 uses a three-dimensional textured model.

Once this two-level classification of the scene is given, it is up to the visualization techniques to determine the appropriate handling of scene objects. The following sections within this chapter deal with this issue in different manners, but they all share the same principle: That the context part of the scene is a resource to be handled carefully rather than disregarded. By following this principle, the perception of depth in mixed reality scenes is improved, as we demonstrate in the following examples.

4.2 The general algorithm

We now outline the general technique used throughout this chapter. The key idea of this algorithm is to extract information from the scene's context. Then, the information is stylized in a way that improves the perception of depth. The main differences between all techniques are how the importance information is extracted and how it is rendered. Algorithm 3 describes this procedure.

```
For every object in the scene do
     Classify as either focus or context, i.e., belonging to either
     f(scene) or c(scene)
End for

For every object in c(scene) do
     Gather the importance information f(c(scene)
End for

Render video background
Render virtual objects
Render stylized form of f(c(scene), i.e., the importance information
```

Algorithm 3. Improve the perception of depth by context preservation. This algorithm shows the basic concept that will be used throughout this chapter.

Steps for gathering the importance information and stylized it vary from technique to technique. For example, Section 4.5.2 gathers the importance information by procedurally generating it upon runtime, while Section 4.4 extracts this information from the video feed. Steps 7-9 are the synthesis steps also sometimes referred to as the compositing steps. Compositing typically involves a complex blending of objects where not only fragment data is considered but also depth information. Kalkofen et al. (Kalkofen, Mendez, and Schmalstieg 2007) provide a thorough description of such a technique.

The final importance information produced by each technique should follow certain guidelines. Interrante et al. (Interrante, Fuchs, and Pizer 1997) suggest that silhouettes and contours provide important cues of the shape of occluding objects:

> *Silhouettes are important for form perception because they define the boundary between figure and ground, and contours, defined as the locus of all points where the surface normal is orthogonal to the viewing direction, mark both internal and external depth discontinuities.*

Principal directions and principal curvatures may also be preserved. Rusinkiewicz et al. (Rusinkiewicz et al. 2008) provided a thorough description of line drawings that may also be used for extracting features such as ridges and valleys. The disadvantage of these techniques

is that they require a virtual model linked to the occluding object. Moreover, these are merely guidelines that can help improve the perception of depth but can be ignored. Notably, the last technique presented in this chapter breaks away from these guidelines with encouraging results.

4.3 Using context from linked virtual models

The first technique we present uses virtual models as a hint for the location of an occluding object. The first requirement is that there exists a virtual untextured three-dimensional model that is a faithful representation of the occluding object. We will refer to this model as the *virtual occluding model*. Notice that this is not the importance information, but will be used as a prop to derive it. This model must be tracked in such a way that it reflects the location of the occluding object.

4.3.1 Gather the importance information

To gather the importance information the object must be rendered to an intermediate texture for later use. This model is not displayed in the final screen, however. We have used the OpenGL extension for frame buffer objects to render directly to a screen aligned texture; alternatively, pixel buffer objects can also be used. Subsequently, a post-processing step is necessary to extract features from the intermediate texture. What these features are depends on the application programmer or the current task at hand.

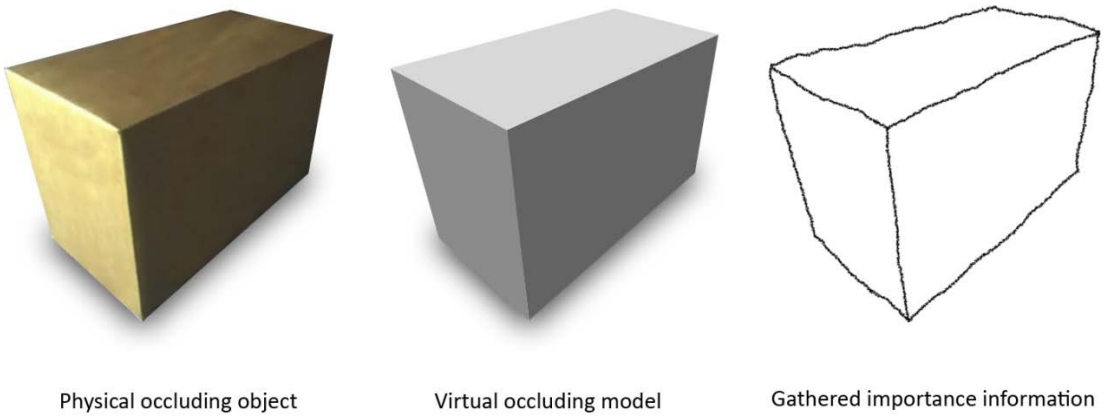


Figure 48. Gathering importance from linked models. (Left) The box represents the physical occluding object in our scenario. (Middle) A linked virtual occluding model, a faithful representation of the physical box. (Right) The result of our importance gathering technique.

However, the feature extractor must preserve certain information from the occluding object, as suggested in Section 4.2. Nevertheless, extracting information, such as silhouettes, is not

straightforward, since the virtual occluding model must be stored in a form that allows the computation of curvatures. For this matter, we opted for extracting features in image space. Such techniques are more sensitive to lighting artifacts, but at the same time are easier to implement and do not require specific data storage. A saliency operator such as that of Achanta et al. (Achanta et al. 2008) or an edge detector such as that suggested by Canny (Canny 1986) are adequate candidates, though more advanced techniques may also be used. In this section, we use the computational edge detector suggested by Canny and we use GLSL fragment shaders to compute it (Rost 2004).

The algorithm for gathering the importance information is shown in Figure 48 and described in Algorithm 4.

```
Disable rendering to display screen
Enable rendering to texture by frame buffer object
Set rendering to Gouraud shading
Render the virtual occluding model
Disable rendering to frame buffer object
Bind the frame buffer object texture for usage
Enable rendering to final importance texture by frame buffer object
Disable depth buffer testing
Bind the Canny edge detector fragment shader
Render a screen aligned orthographic quad
Enable depth buffer testing
Disable rendering to frame buffer object
Enable rendering to display screen
```

Algorithm 4. Context from linked virtual models. This algorithm describes the procedure for context preservation from a linked virtual model.

The output of this algorithm is then the importance texture that is used for the next step. Notice that implementation details are left out; for example, the Canny edge detector is actually a multi-pass rendering technique and not a single pass as described here. We chose this edge detector for its relatively easy implementation; however, the algorithm allows the usage of other detection techniques.

We used Gouraud shading (Gouraud 1971) for two reasons: 1) it is implemented in hardware, and 2) it reduces the amount of shading discontinuity that flat shading produces. However, the technique is not bound to this particular shading technique, and other alternatives, such as Phong shading (Phong 1975) may be used.

Figure 48 shows a real occluding object, its associated virtual occluding model, and an illustration of the result of the edge detector. After computing the importance information, which in this case tends to be high curvature features, we can proceed to stylize this information for actual display.

4.3.2 Render a stylized form of the importance information

Having collected the importance information, we proceed to present it to the user. This section, in particular, describes our first attempt at depth perception enhancement. Hence, the resulting importance information is presented to the user in an abstract form. The resulting importance values are mapped to a specific color chosen by the programmer during the design of the application. In the general case, this color mapping should not be hard coded but should depend on the underlying video information. Chapter 5 suggests a solution to this problem in the general framework of attention direction.

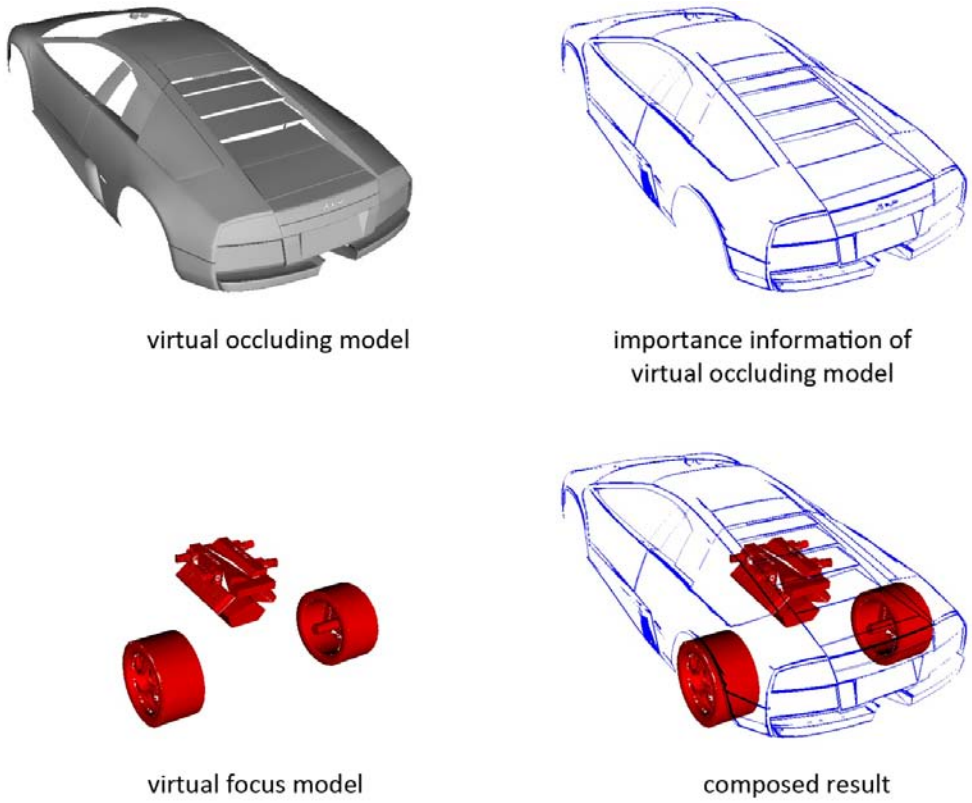


Figure 49. Steps for the synthesis of the final image. (Top left) Rendered version of the virtual occluding model. (Top right) Result of the Canny edge detector that, for illustration purposes, is shown in blue. (Bottom left) Virtual focus object. (Bottom right) Composed result.

Figure 49 illustrates the steps necessary to synthesize the final image as described by Algorithm 4. First the virtual occluding model is rendered in gray scale with Gouraud shading (Gouraud 1971). Then we post-process it with an edge detector (a Canny edge detector in this instance). The virtual focus object (engine and back tires in this case) is rendered without any type of pre or post-processing. Finally, the synthesized image is simply the overlay of the detected edges over the virtual focus.

To demonstrate an example of this, we created a small AR application. This consisted of a physical car model, a virtual engine and tires of the car as focus objects. And finally, a faithful

three-dimensional model of the physical car is used as the virtual occluding model. This application simply maps the importance to a gray color. Notice that in the illustration shown in Figure 49, we used a blue color for the importance for the purpose of clarity. Throughout this section, the choice of which color to use depends on the application programmer. Figure 50 shows a crude augmentation of a car engine and its back tires on top of a physical car model. On the bottom, our context preservation technique is shown where the importance information, extracted from an edge detection of the rendered virtual occluding model, is overlaid on top of the focus.

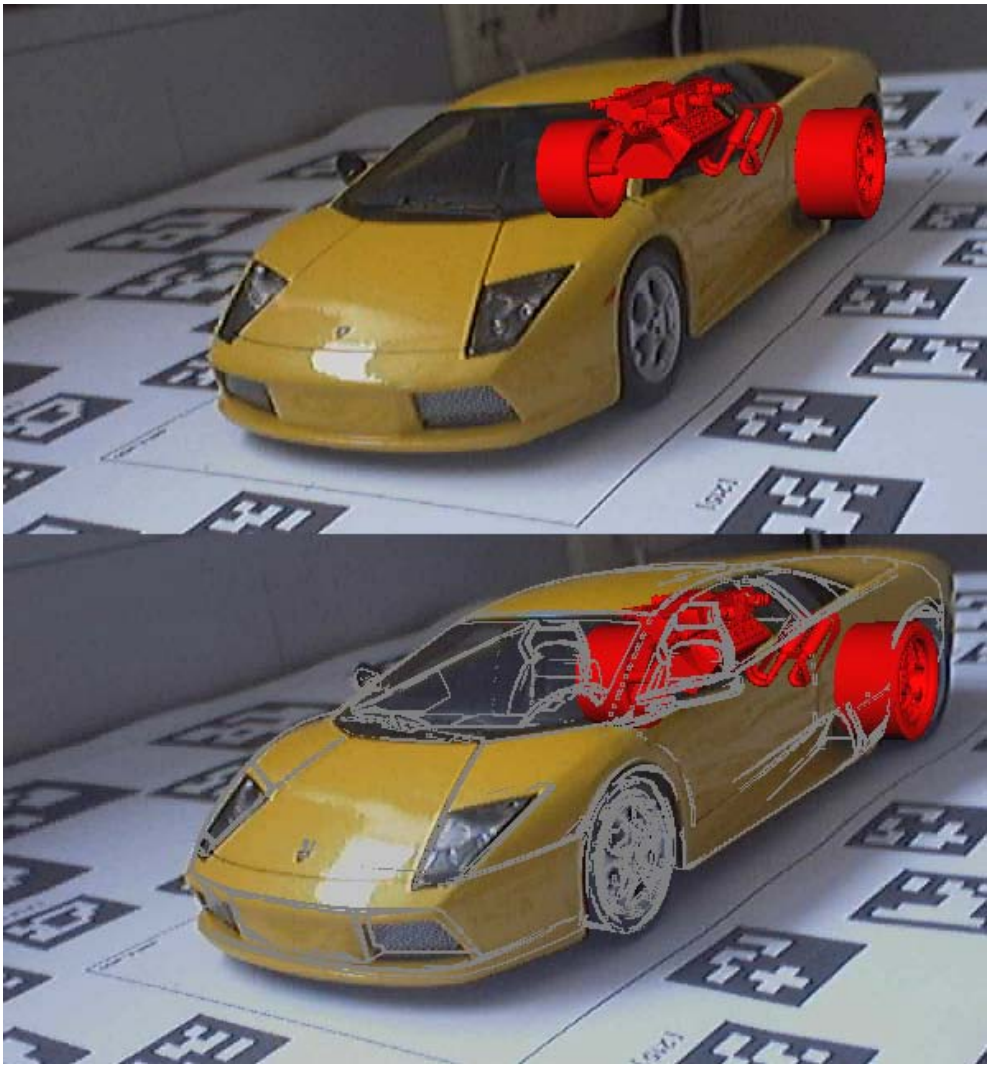


Figure 50. Using context from linked virtual models. (Top) This image depicts a crude augmentation of the car engine and back tires. (Bottom) This image shows that one can improve the perception of depth by adding hints at the locations of the important features of the context.

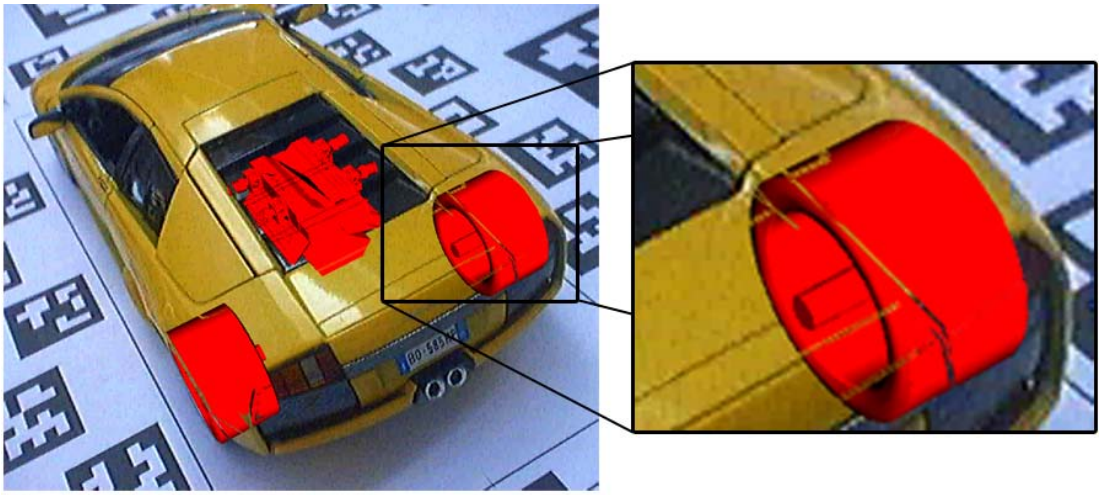


Figure 51. Edge-preserving video. This figure illustrates the problem of not using an abstract representation of the importance information. In this example the extracted edges are used to preserve data from the video. Unfortunately, this technique achieves little contrast and is of little use.

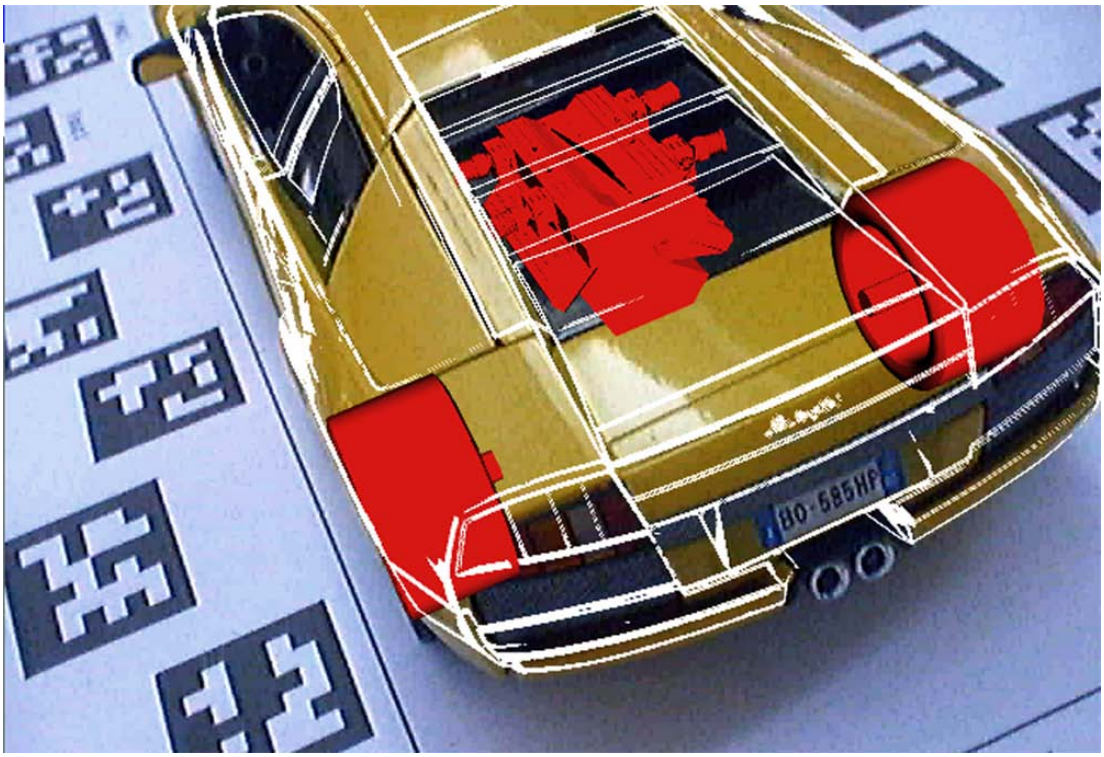


Figure 52. Better context preservation. This image illustrates how higher contrasting edges better enhance the perception of depth. Compare this image with the result of Figure 51.

4.3.3 Edge styling

In this section, we put forward the concept of presenting the importance information in an abstract form. One alternative to this is to use the importance information as a mask to preserve the contextual information; this idea is explained in detail in Sections 4.4 and 4.5. However, the problem of preserving the raw detected edges is that the information preserved offers little contrast to the augmentations. Thus, in the resulting image, the preserved edges are barely visible. Such sparse representations provide only a small number of features, making it difficult to mentally reconstruct the object's shape, especially if its appearance is similar to its surroundings. Figure 51 illustrates this problem. Notice in the zoomed image on the right that the edges indeed preserve information from the video feed, but are so thin and inconspicuous that the perception of depth remains poor. Figure 52 shows a solution with higher contrasting edges.

Another technique to communicate spatial arrangements of such representations is edge stippling, or edge styling. This technique creates discontinuities directly on an edge, rather than on intersections among edges. Styling is a widely known convention in illustration that conveys the spatial relationship among objects (Appel, Rohlf, and Stein 1979; Kamada and Kawai 1987). For example, edges from back or inner faces are indicated by dotted edges, while front facing or occluding edges are continuous. Given a representation with reasonable straight lines, styling provides additional depth cues without consuming additional image space. However, the use of stylized edges requires a limited amount of image clutter, as well as a limited number of depth levels, to be effective. For example, small halos around the edges provide the user with occlusion cues among the edges themselves as suggested by Appel et al. (Appel, Rohlf, and Stein 1979). While edges at the front appear as straight lines, edges at the back become less dominant near foreground edges. Figure 53 shows three examples of edge haloing techniques with detail insets on the right. Figure 53 (top) shows edges overlaid without haloing. Figure 53 (middle) shows edges with hard halos. Notice that the edges are terminated rather abruptly. Figure 53 (bottom) shows edges with smoothed halos. Notice that the final result is slightly more pleasant to look at. Unfortunately, image-based edge detectors may compute many disconnected lines. Adding more discontinuity with hard edge halos, may exacerbate the image clutter, although this can be minimized with continuously transparent halos.

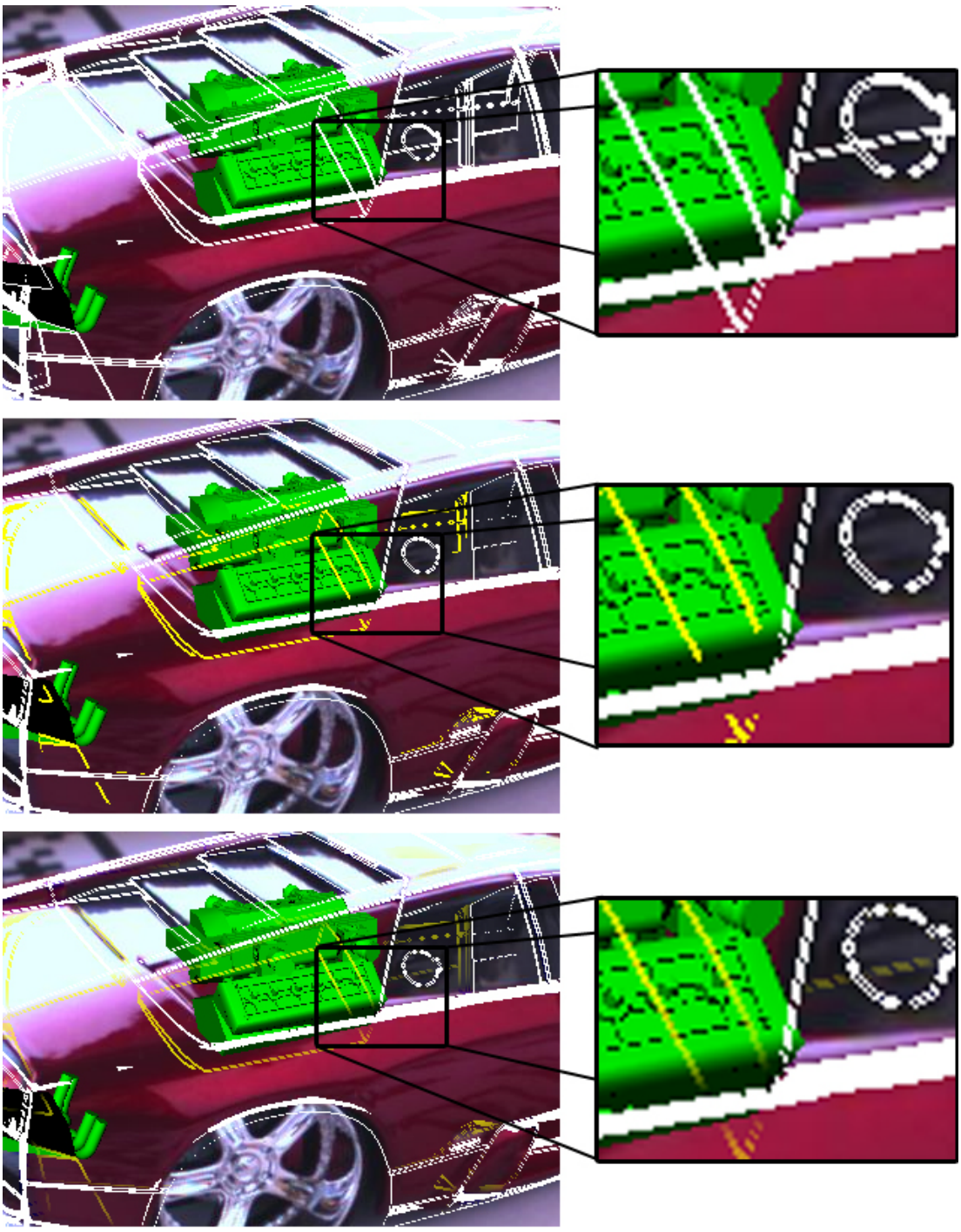


Figure 53. Sketch-inspired edge preservation. Each approach is shown with a detail inset on the right. (Top) Original edge preservation presented in this section. (Middle) An example of inter occlusions between internal edges (yellow) and external (white). (Bottom) Internal edges progressively fading as they approach an external edge.

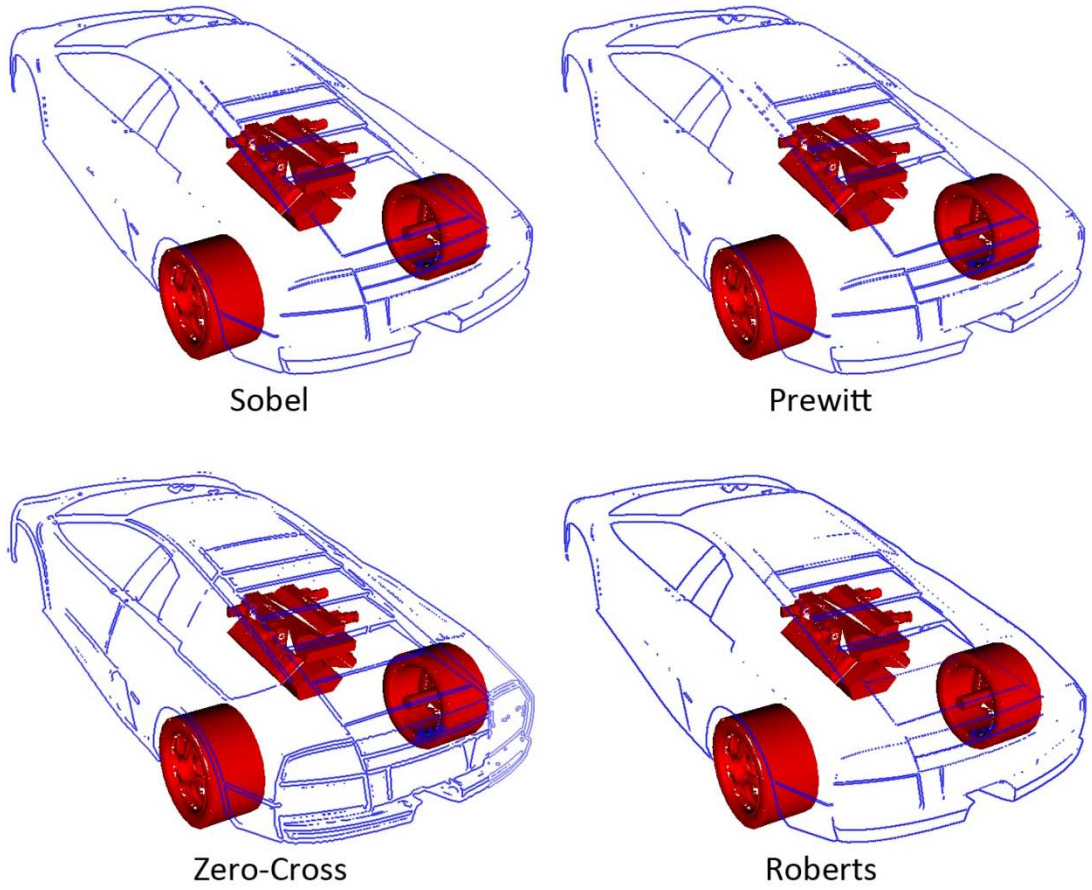


Figure 54. Multiple edge detection strategies. (Top left) Sobel edge detector. (Top right) Prewitt edge detector. (Bottom left). Zero-cross edge detector. (Bottom right) Roberts edge detector. These example images were constructed in Matlab for illustration purposes.

4.3.4 Discussion

We have shown a first implementation for depth perception enhancement based on linked virtual models. Of all the techniques presented in this chapter, this is the only one that does not truly preserve context information; instead, it adds information that hints at the location of the context features. This technique uses a virtual occluding model bound to the location of the physical occluding object. The hints added to the final image are derived from analyzing a rendered image of the virtual occluding model.

As can be seen from Figure 50, our technique is capable of enhancing the perception of depth of single layer occlusions. Notice that the color chosen for the hints is contrasting enough to the video feed, but not visually disturbing. The color used is chosen manually to achieve these effects; in practice, automated analysis of the image would be desirable.

The choice of the particular engines used to generate the images in this section should be revisited since there exists much work on line stylization on the NPR community. The choice of the Canny edge detector was based only on its ease of implementation; in contrast, the

next section, for example, instead uses image variance as the edge detector. Figure 54 compares multiple edge detectors.

Section 4.3.3 used the existing technique of "interrupting" edges that intersect others (Appel 1967; Johnson 1963). To accomplish this, one would ideally want to use the distance transform (Danielsson 1980). The distance transform would provide an exact description of the distance between every pixel and the nearest edge. This would enable a more accurate blending between occluding and hidden edges. However, when we developed our implementation, there was no fast GPU implementation of the transform that could handle interactive frame rates. Thus, the use of the haloing technique was born out of necessity since it visually resembles a distance transform yet it can have a faster implementation. The haloing technique used was that proposed by Bruckner and Groeller (Bruckner and Groeller 2007). A better technique would now use a GPU implementation of the distance transform such as that presented by Rong and Tan (Rong and Tan 2006).

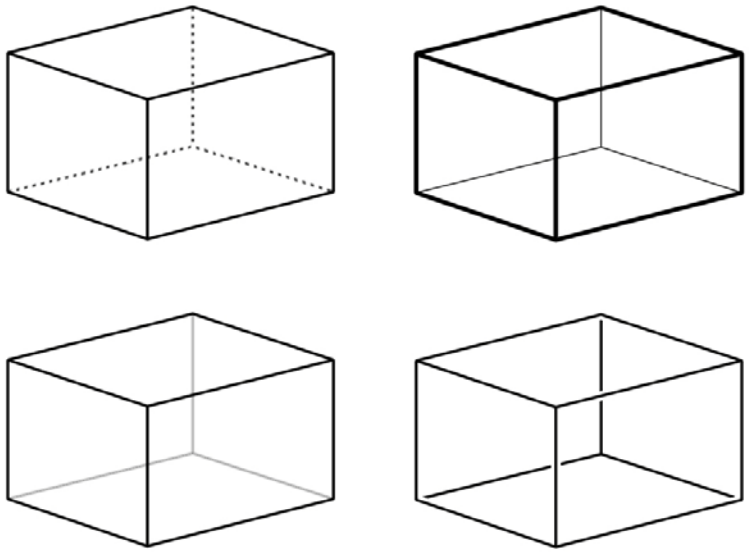


Figure 55. Edge stylization strategies. (Top left) Back edges are dashed. (Top right) Back edges are thinner. (Bottom left) Back edges are lighter. (Bottom right) Edges are haloed.

There exist several strategies to stylize the edges. Drawing artists have since long used the stylization of edges to convey distance. For example, thinner, dashed, dotted or light colored edges all imply a far distance to the viewer. Figure 55 illustrates several edge stylization examples. In our work, we have only experimented with the haloed edges (Figure 53).

The edges throughout this section are always extracted from a two-dimensional rendered image of the three-dimensional model. The result is thus dependent on the rendering technique used. As mentioned before, we used Gouraud shading (Gouraud 1971), as it proved suitable to our needs. Different shading techniques such as Phong (Phong 1975) or flat shad-

ing can give drastically different results. The position of lights, the angle of the camera, shadows and textures will also influence the final result. A better technique would then use edges extracted from geometry instead of the rendered model. These edges can then be the source of information such as curvature, occluding contours (Koenderink and van Doorn 1998), suggestive contours (DeCarlo et al. 2003), ridges and valleys (Ma and Interrante 1997), isophotes (Goodwin, Vollick, and Hertzmann 2007) or apparent ridges (Judd and Durand 2007).

The next section will continue to extract the edges from a two-dimensional image. However, the image will be the video image itself and not a three-dimensional model. The styling will not be synthetic, but will instead use the video itself. It will also continue to use the haloing technique as a substitute for the distance transform.

4.4 Using context from inferred information from the video feed

The technique described in the previous section assumed that the source of contextual information is a linked virtual occluding model. However, another important source for AR scenes is the video stream used for video see-through augmentation. A video stream delivered as a texture can be subjected to all image-based operations described earlier. Techniques presented in Sections 4.3 and 4.5 require the existence of a three-dimensional model as a linked virtual occluding model. In contrast, the technique presented here does not have this prerequisite, but is based on the assumption that all virtual focus objects lie behind the video feed. Unlike the other techniques, this section cannot deal with only partially occluding objects. Nevertheless, many situations in AR applications allow for this setup to be used.

4.4.1 Gather the importance information

In Section 4.3 gathering the importance information is done by applying an image analysis of the rendered virtual occluding model. The result is thus dependent on the quality of the tracking and registration of the virtual model. Figure 56 (top) illustrates this problem, where the rendered model is not well registered with the physical counterpart. What we propose here, is that this problem may be overcome if we rely on image analysis of the video stream instead. Figure 56 (middle) shows an example of this. Notice that the edges are perfectly registered, but rather thick and less detailed. The quality of the edges depends not only on the edge extraction technique itself, but also on the quality of the video image. Most of the time, edges extracted from a rendered model will have superior quality to those extracted from video. Of more concern is that edges extracted from video may clutter the whole view. This can, however, easily be solved by limiting edge detection to the footprint of the rendered model (Figure 56 bottom).

A better image analysis strategy may be used, but the results will always depend on the quality of the camera feed and the current illumination conditions. Section 4.3 used a Canny edge detector for the analysis of the rendered virtual occluding model. The Canny edge detector demands multiple passes on the image before the final edges are obtained, which is computationally expensive. Instead, in this section, we use image variance to obtain the edges of the video. The resulting edges are cleaner and better registered to the video information and the implementation can be done fast in a single pass.

The problem remains, nevertheless, that the detected edges are far too sparse for our purposes (see Section 4.3.3). To alleviate this, we would ideally compute a distance transform on the edges of the video such as suggested in Section 4.3.4. Figure 57 illustrates the results of edges detected with the image variance (middle) and after haloing (bottom). Notice that the mask is actually inverted for readability.

4.4.2 Render a stylized form of the importance information

Adding extra artificial information to the image may be distracting and increase image clutter (such as the yellow edges of the images in Figure 56). A more subtle way to depict context is by using the original video information as context overlay. The limits of human perception impose a trade-off between the number of preserved features and the clarity of revealed hidden structures. If the visualization demands a high clarity of hidden structures, a sparse representation has to be used for occluding objects. However, this sparse representation may also be problematic, as discussed in Section 4.3.3. The visualization should aim then to contribute features of optimal clarity. In case of sparse preservations, such as those of Figure 51, the features are more comprehensive if they have a consistent tone or color and stand in strong contrast to their surroundings. Unfortunately, video-based AR does not guarantee that such features exist in the original video.

What we propose then, is to preserve information from the video itself, rather than adding extra augmentations for context preservation. This has several advantages: image clutter is reduced, rather than increased, and the approach is insensitive to registration and tracking errors. The construction of this stylized form is simple. First, we mask the incoming video image using the haloed edges as alpha values. Next, we render the elements in the following order: 1) video feed, 2) augmentations, and 3) masked video. Figure 58 illustrates this process.

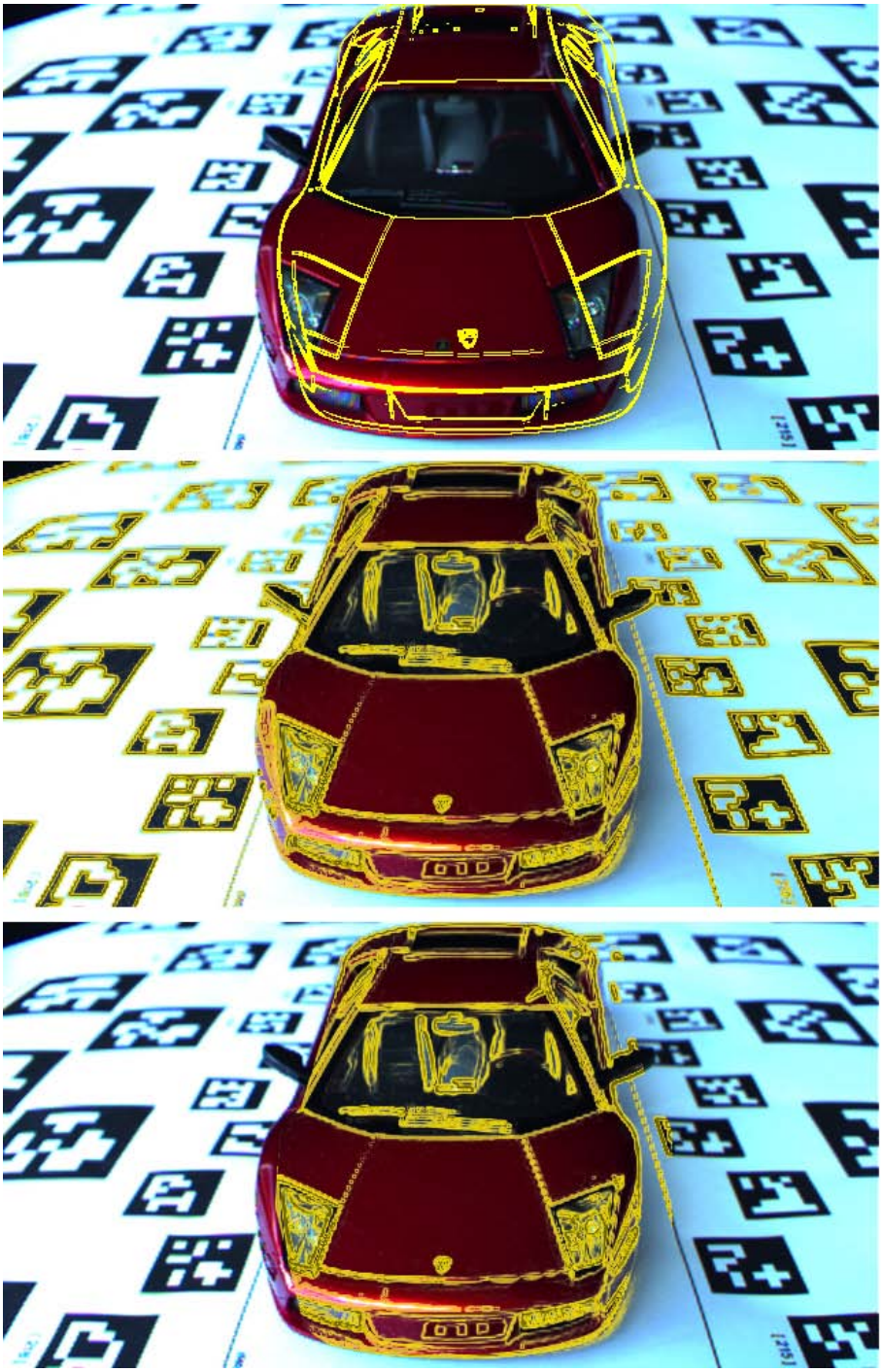


Figure 56. Model-based vs. image-based edge detection. (Top) Edges are clear and well defined but suffer from tracking and registration errors. (Middle) Edges are perfectly registered to the video but are noisier, less clear and cause more clutter. (Bottom) Edge detection is restricted to those falling inside the rendered model, thus reducing clutter.

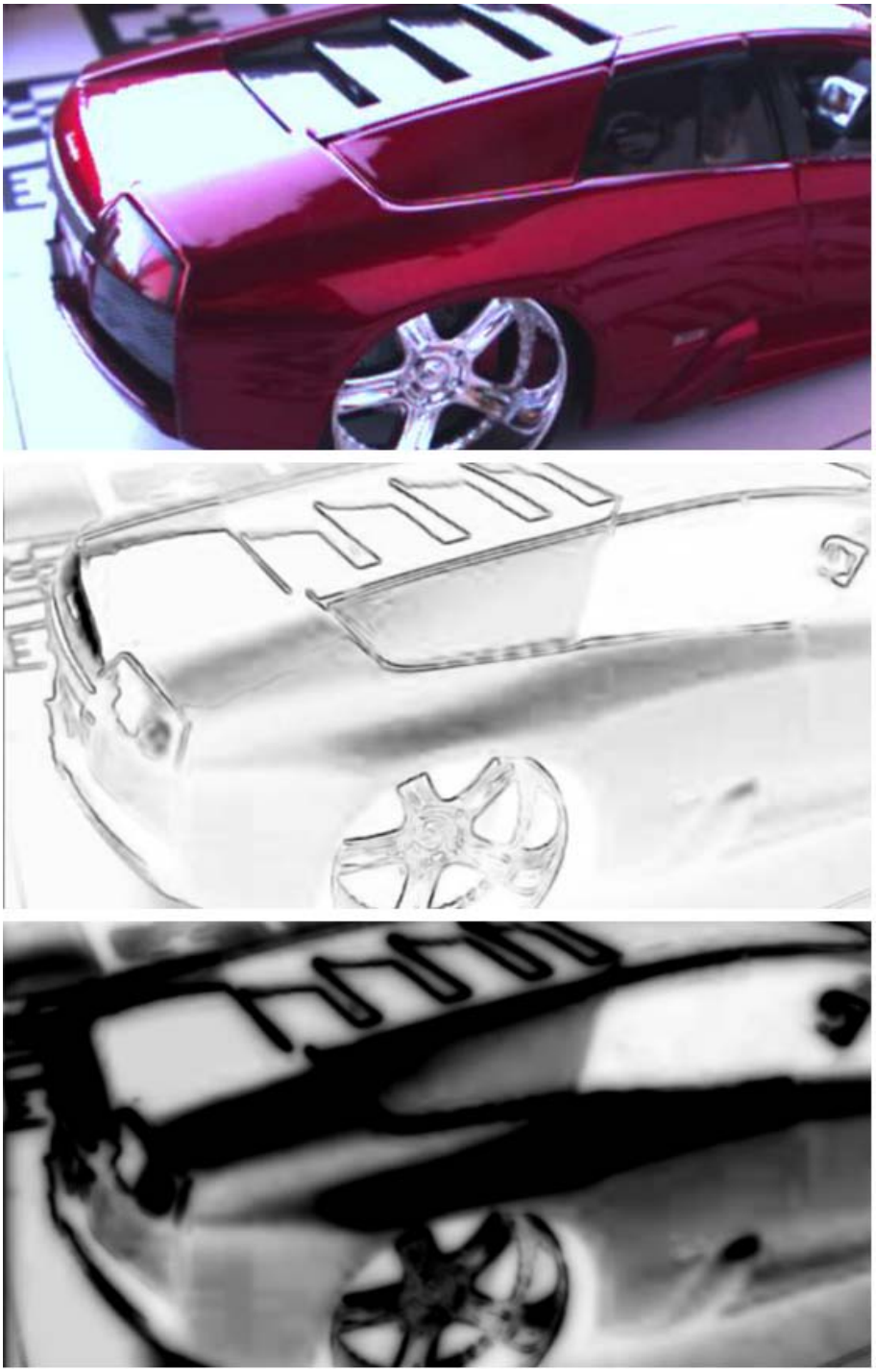


Figure 57. Haloed edges from variance. (Top) Original image. (Middle) Resulting edges (in black) from the image variance. (Bottom) Edges after haloing with the technique by Bruckner and Groeller (Bruckner and Groeller 2007).

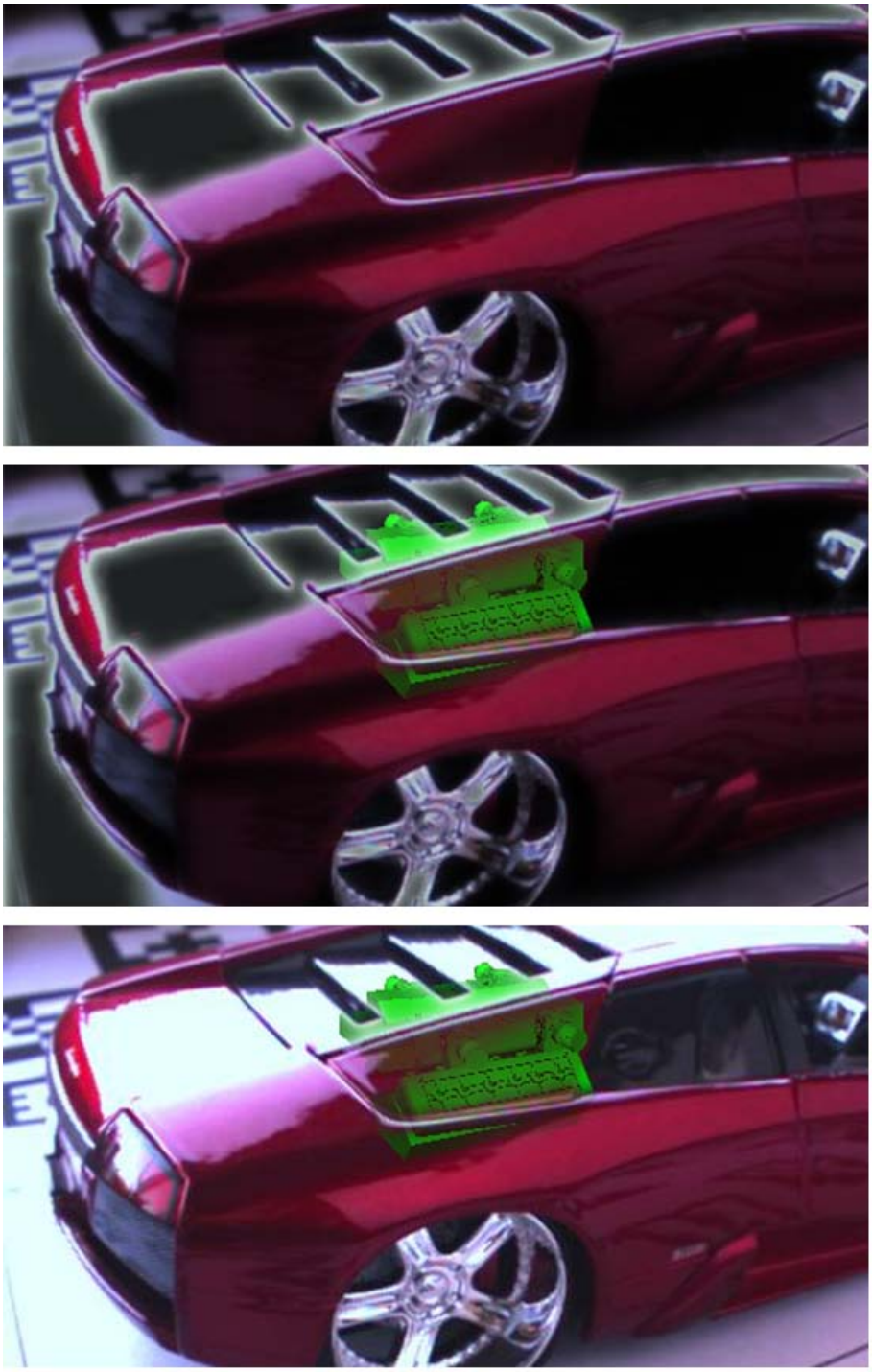


Figure 58. Constructing the stylized rendering. (Top) Input video masked with the haloed edges from Figure 57. (Middle) Masked video rendered on top of the augmentation. (Bottom) Overlaying the stylized rendering on top of the augmentation and the video feed to obtain the resulting image. Notice that no extra cluttering artifacts are introduced and that the stylized rendering is perfectly registered with the video feed.

Figure 59. Visual results of preserving context from video. The resulting images are visually pleasing, although they are sensitive to lighting changes and camera angles.

4.4.3 Discussion

We have presented the second technique in this chapter for enhancing the perception of depth with context preservation. This technique has a different set of assumptions than that presented in Section 4.3. For example, augmentations in this technique are assumed to be always behind the video feed. The resulting images are more visually pleasing as they contain less image clutter and are better registered with the scene. Figure 59 shows two images rendered with this technique. Notice that the resulting images are dependent on lighting changes and camera angles. This causes certain artifacts such as shimmering edges on shiny objects.

Additionally, situations may occur when structure to extract the edges is far too scarce (such as a flat painted wall). This is, however, dependent on the quality of the capturing device. In reality, it is hard to find an object that does not possess some sort of texture information, albeit minimal. In current AR applications the video camera used is typically incapable of perceiving such small detail. In such situations the current technique would fail to convey enough structure to communicate the order of the objects. The next section will deal with this problem with two different strategies: predefined and procedurally generated.

4.5 Using context with a predefined or procedural mask

The previous sections are based on the concept that the importance mask should be generated on a frame-by-frame basis. In Section 4.3, we applied an edge detector to a linked virtual occluding model, while in Section 4.4 an edge detector and a haloing effect are applied on the video layer. Both techniques, however, rely on the assumption that the linked virtual model or the camera image possesses enough information so that the detection can preserve sufficient contextual hints; however, this might not always be the case. This section proposes two ideas to solve this problem, first, to predefine a mask mapped to the virtual occluding model, and second, to procedurally generate this mask during runtime. By doing this, we restrict the number of occluding objects that can be used and they must necessarily be static or tracked. Nevertheless, the generation of the resulting image is less computationally expensive and is not subject to the pitfalls presented in the earlier sections.

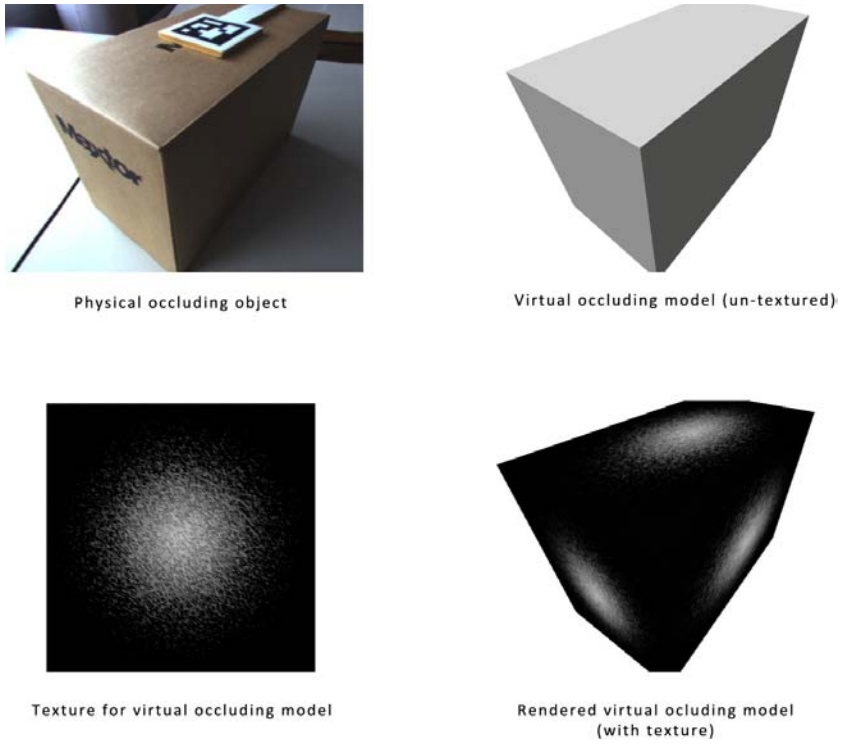


Figure 60. Virtual and physical actors of the predefined mask. (Top left) Physical occluding object. (Top right) An untextured version of its virtual occluding model. (Bottom left) Texture that is used in all faces of the virtual occluding model. (Bottom right) Rendering of the textured virtual occluding model.

```
void main(void)
{
    vec4 Tf = texture2DRect(tex_focus, gl_FragCoord.xy).rgba;
    vec4 Tm = texture2DRect(tex_importance, gl_FragCoord.xy).rgba;
    vec4 Tv = texture2DRect(tex_video, gl_FragCoord.xy).rgba;
    if (Tf.a==0.0)
        gl_FragData[0].rgb= Tv.rgb;
    else if (Tm.a==0.0)
        gl_FragData[0].rgb= Tv.rgb;
    else
        gl_FragData[0].rgb=( Tv.rgb* Tm.a)+( Tf.rgb*(1.0- Tm.a));
}
```

Algorithm 5. Pseudo-code for importance masks. This pseudo-code in GLSL format illustrates the procedure for context preservation from predefined masks.

4.5.1 Predefining a mask

Similar to the technique presented in Section 4.3, this technique requires a model of the occluding object. This model must, however, be textured. The virtual occluding model must also be tracked in order to reflect the location of the physical occluding object. The texture in the general case should be gray scale and it should convey the amount of preservation we want to achieve. This is convenient, as we can then custom define exactly what portions of the occluding object convey better the shape without relying on an online analysis.

4.5.1.1 Gather the importance information

Similar to Section 4.3, to gather the importance information the textured model must be rendered to an intermediate texture for later use. This model is not displayed in the final screen, however. We again use the OpenGL extension for frame buffer objects to render directly to a screen-aligned texture. Unlike the technique presented in Section 4.3, we do not present the importance in an abstract form, but instead use it as a mask for the video feed, similar to Section 4.4. Once rendered, the resulting texture tells the system which fragments from the incoming video input to blend with the virtual focus model.

Figure 60 presents a physical object that will act as an occluding object in our example. It illustrates a photograph of the physical object and all the components necessary for the definition of the textured virtual occluding model. The final rendered image is shown on the bottom right. This result is written to a texture as the final importance mask.

The predefined mask (Figure 60, lower left) uses black to indicate regions of high importance (meaning, “no go” areas) and white to indicate regions of low importance. In this case, the borders of the texture are highly important, smoothly decreasing towards the center. This follows the suggestion of Interrante et al. to preserve the contours of the object. Moreo-

ver, the gradient suggests curvature at the edges. The sketchy information in the center compensates for the lack of structure in the center of the box.

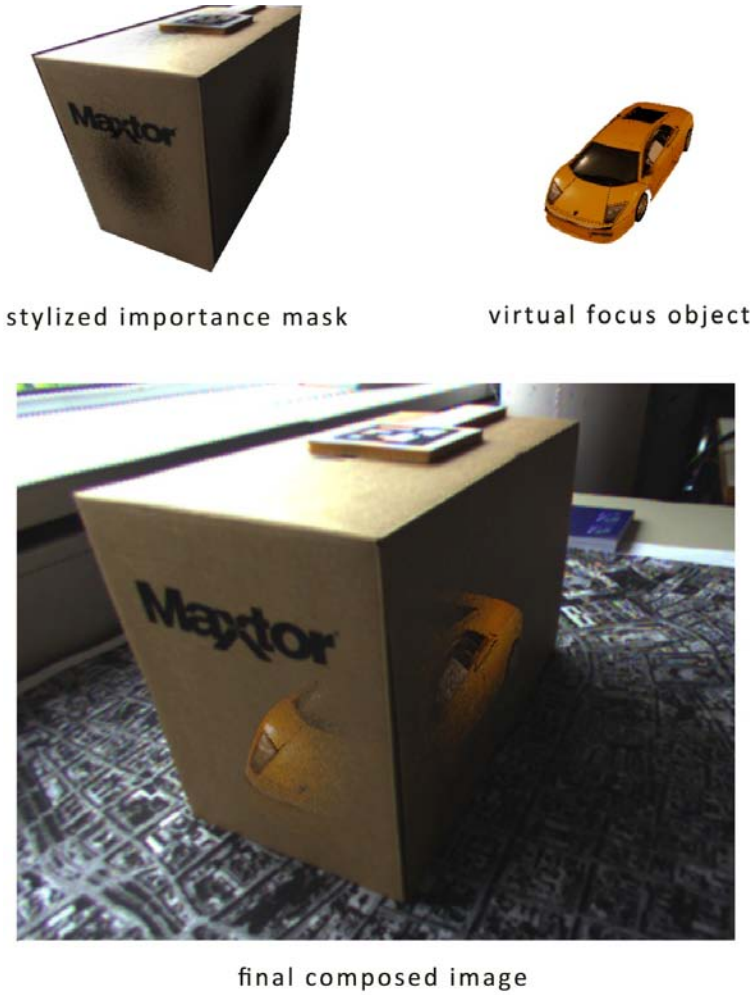


Figure 61. Example usage of the procedural mask. (Top left) Stylized virtual occluding model. (Top right) A three-dimensional model that represents the physical occluding object. (Bottom) The final synthesized result of this technique.

4.5.1.2 Render a stylized form of the importance information

Having gathered the importance information, we proceed to synthesize the final resulting image. We have already presented in Section 4.4 a comprehensive algorithm for how to use the importance mask to blend video information on top of the virtual focus. However, a small GLSL pseudo-code fragment shader is presented Algorithm 5 for convenience.

We assume as given the textures for the focus object, *tex_focus*, the importance mask, *tex_importance*, and the video feed, *tex_video*. We assign the current fragment values to T_f, T_m and T_v, respectively. The blending of the video and the focus is driven by the values in the importance mask. It is important to remember that this technique is specific for single layer

occlusions and that, in the general case, it will fail or provide inadequate results if more occlusion layers are used.

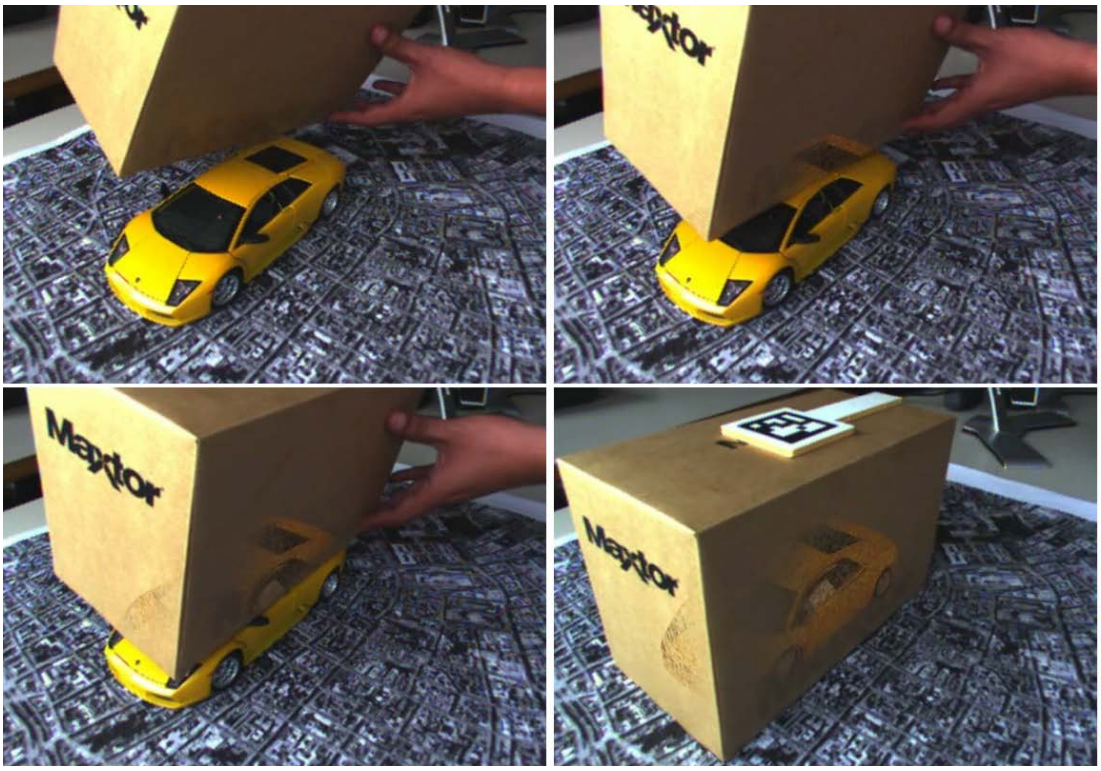


Figure 62. Occlusion sequence. This four-panel figure illustrates the effect achieved by our technique when the focus of interest (the car) is occluded by the context (the box).

Figure 63. Outdoor example of the predefined mask. This figure illustrates the effect of our technique when trying to visualize the inside of a building. The predefined mask, in this case, preserves the edges of the windows.

We now present a small example of this technique based on the sketch presented in Figure 41 and Figure 42. Assume that we want to see the insides of a box that contains a model car.

Figure 61 illustrates all the actors of this technique. On the top, we can see the stylized importance mask, the virtual focus (representation of the hidden physical object). On the bottom, we show the final synthesized result. This particular image preserves a large amount of the occluding object's information. Thus, a tradeoff exists between the amount of preserved information and the gap left for the virtual focus to be shown. That tradeoff is a research direction that needs to be explored.

Figure 62 and Figure 63 illustrate two more examples of our predefined mask technique. Figure 62 shows a sequence of frame grabs from a video in which we occlude the object of interest (the car) with a physical object (the box). This mask preserves the edges as well as adds extra structure to the sides of the box; notice that it is the same mask constructed in Figure 60 and Figure 61. Figure 63 shows an outdoor application. In this scenario, the user stands outside a building and wishes to see the inside of an office. Notice how the mask preserves the metallic frames of the windows.

4.5.2 Procedural mask

Predefining an importance mask gives the application designers complete control over which features of the occluding object are important and which are not. However, a strong pre-requisite is that there needs to be previous knowledge about which objects are to be considered as occluding objects. A second prerequisite is that the importance mask must be somehow pre-generated, potentially following guidelines such as those given by Interrante et al. (Interrante, Fuchs, and Pizer 1997). This section explores a different direction, in which the importance mask is procedurally generated, instead of being predefined or generated based on image analysis.

4.5.2.1 Generating the importance information

Unlike the previous sections of this chapter, this technique does not gather the importance information, but generates it at runtime. Similar to Sections 4.3 and 4.5.1, this technique requires the definition of the occluding object in advance. Unlike other techniques, this technique requires a mask-generating engine. Additionally, the model may require texture-coordinate mapping, depending on the particular engine.

The generating engine is a little program that, upon runtime, dictates which portions of the image should be preserved and which may be ignored. The complexity of this engine may vary from simple monotonic checkerboard generators to highly complex engines that consider the curvature of the virtual occluding model. In this section, the engine takes the form of a fragment and a vertex shader. We have created an engine that uses a variant of the Perlin turbulence function (Perlin 1985). This engine does not consider the geometric information of the virtual occluding model; thus, no edges or other high curvature features are pre-

served. The technique presented here is a deviation from previous work in this chapter, as it does not follow any of the guidelines of Interrante et al. (Interrante, Fuchs, and Pizer 1997). Instead, it generates the mask pseudo-randomly for each location in the texture. This is not a limitation of the concept of procedural generation of textures, but of the particular engine we used. Nevertheless, this technique has its advantages; for example, the technique is independent of the structural information of the physical occluding object. In other words, whether the physical object has enough structure for preservation (necessary in Sections 4.3 and 4.4) is irrelevant. Thus, objects that offer little to no structure for preservation (such as tables or walls) are handled well.

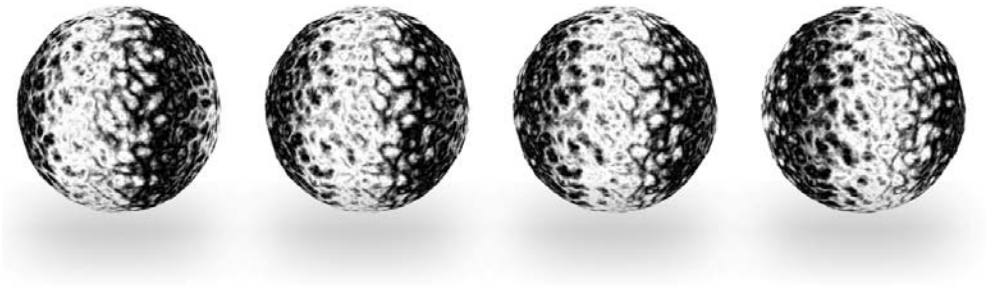


Figure 64. Illustration of the phase parameter. These four images show an object textured with the marble texture with a slightly different phase value. In our prototype, the phase value is bound to the angular position around the *Y* axis of the camera, thereby creating a "shower glass effect".

Another advantage of this technique is that the importance information is not bound to the underlying physical object. This enables it to vary the resulting importance mask during runtime. For example, the marble texture described by Perlin allows control of the frequency, amplitude and phase of the resulting texture. The parameter frequency controls the number of stripes in the texture, while amplitude controls how distorted the stripes look (see Figure 64). The last parameter, phase, controls a ripple-like distortion, similar to the spatial distortion one sees when looking through shower glass. This last parameter is of interest, as it can be bound to the relative position of the camera to simulate a consistent shower glass effect. Figure 64 shows a series of images with a progressively increasing phase value. In these examples, the object is rotated around *Y* to illustrate the effect of phase. In our prototype, the phase value is bound to the angular position around the *Y* axis of the camera.

4.5.2.2 Render a stylized form of the importance information

Once the importance information has been generated, we proceed to synthesize the image in a way similar to that of Section 4.5.1. The results are hard to convey with static imagery, as they are aided by temporal and view angle changes of the camera. Figure 65 shows an image sequence of the procedural texture based on the marble function. Notice that the edges of the virtual occluding model are disregarded, but no discontinuities are present. Additionally

the virtual back faces of the box are shown in this model (this technique will be described in detail in Section 4.5.3). In this sequence the phase value is animated to follow a sine function of elapsed time.

Procedurally generated images can create visually convincing results, but their effectiveness will depend highly on the generating engine used for the particular application. One can further expand this by introducing different procedural texture generators that consider the topography of the virtual occluding model. Temporal and spatial coherence can also be considered by, for example, adding a viscosity function.

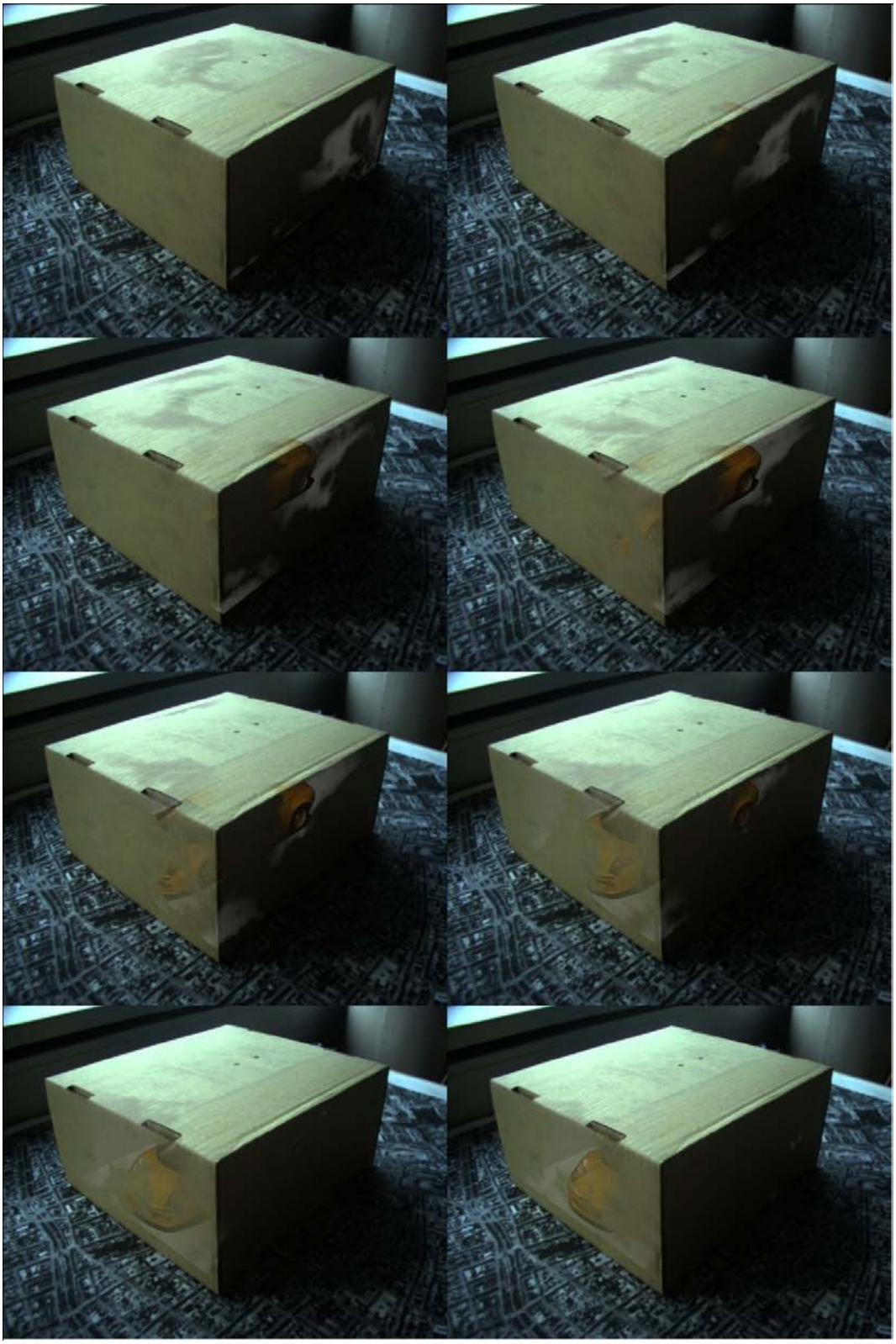


Figure 65. Using a procedural mask. This image sequence shows screenshots from an example application with animated procedural textures. In this case, the phase parameter has been bound to elapsed time.

4.5.3 Expanding the definition of X-ray visualization

Throughout this chapter, we have referred to the problem of conveying spatial arrangements as a depth-perception problem. The solutions that address this problem have been referred to as *occlusion-management techniques*. Other terms to refer to this family of techniques include *information revealing*, *focus and context,* and *X-ray visualization*. Information revealing is perhaps the more appropriate but X-ray visualization is the most widely used term. X-ray visualization is a misuse of the word *X-radiation* that refers to a form of electromagnetic radiation. The meaning used in computer graphics, however, refers to the idea of "Superman's X-Ray Vision" (Livingston et al. 2003), in which the observer is capable of seeing through multiple occluding layers.

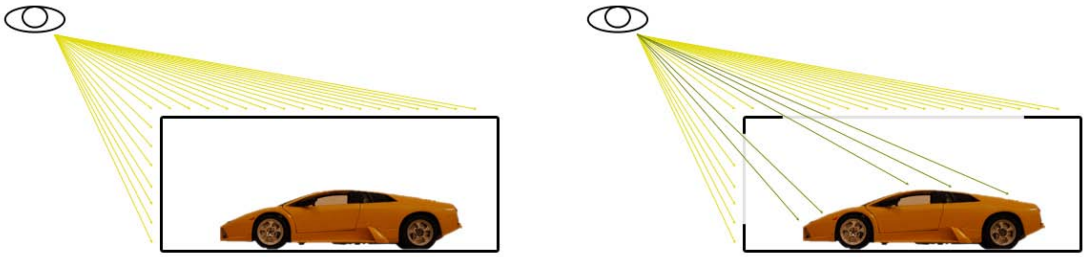


Figure 66. Typical X-ray visualization. (Left) Hidden object encased by a box and an illustration of the camera position. (Right) Conceptual representation of the predefined importance mask presented in this section. Notice that the mask (light gray boundary on the box) extends beyond the focus object yet no extra rays pass through it.

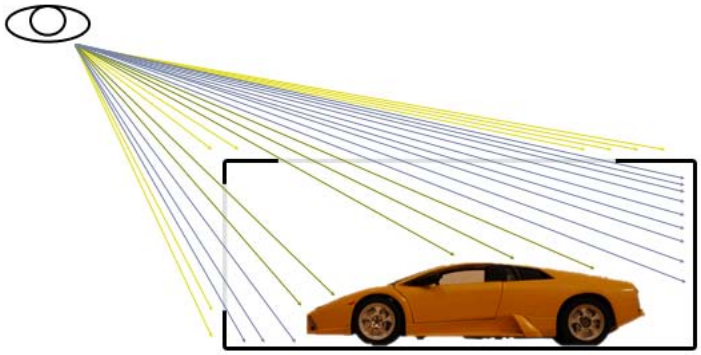


Figure 67. An expanded definition of X-ray visualization. We expand the number of "X-rays" that go through the occluding object (light blue), to all those allowed by the importance mask. These rays "stop" at the moment where they hit another solid object.

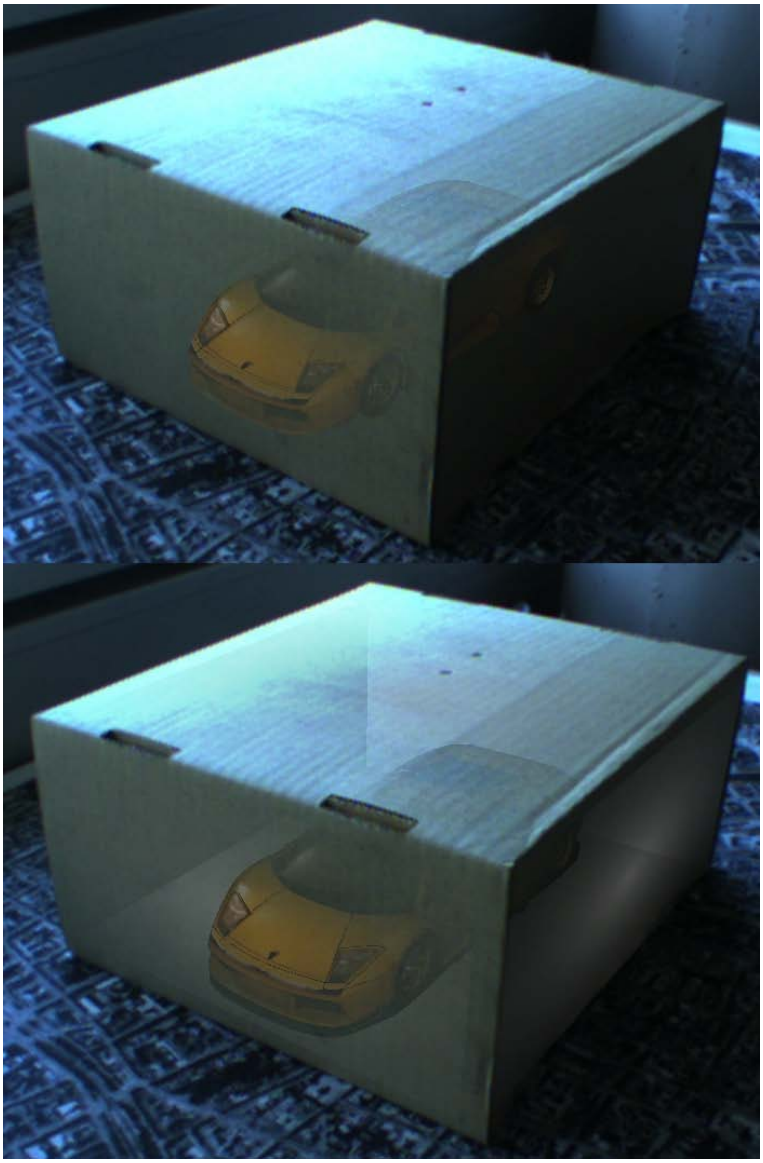


Figure 68. Application of expanded X-ray visualization. (Top) Predefined mask technique presented in Section 4.5.1. (Bottom) Extended definition of X-ray visualization allows to show extra information surrounding the focus, in this case a simple representation of the back faces of the box.

Thus, the techniques presented in this dissertation present a fanciful portrayal of how Superman's X-ray vision works. In this sense, the technique shown in Section 4.3 as well as exploded view diagrams (Li, Agrawala, and Salesin 2004) and cutaways (Feiner and Seligmann 1992) are not physically correct examples of X-ray visualization. X-ray visualization must preserve the spatial arrangements of objects, they must be aligned to the position of the virtual camera and they should not introduce artificial artifacts (such as those shown in Section 4.3). The techniques presented in Sections 4.4 and 4.5 satisfy these constraints by presenting the focus object in its original spatial location and not introducing extra artificial artifacts. However, a special case occurs in which the solely revealed object is the focus of inter-

est. Figure 66 illustrates this situation; notice that on the right image only the rays that hit the focus object go through the occluding object, although the importance mask allows for a higher density of rays to pass.

A better X-ray technique would consider not only the focus object, but also surrounding information. If we were to truly possess (Superman's) X-ray vision, we would see objects in the spatial vicinity of the focus, such as the back faces of the occluding object (another level of context). Figure 67 illustrates this concept. The rays that pass through the occluding object are all those allowed by the mask. In this particular case, they finally hit the back faces of the occluding object, but other information may be revealed instead. Figure 68 shows an actual AR application of this example. Notice that on the top, the focus object seems to be floating inside the box, while on the bottom a better sense of its containment inside the box is conveyed.

This imposes another requirement on the application. If this extended version of X-ray vision is used, the application designer also needs an extended knowledge of the scene. This is similar to the extra model used by the cutaway restrictions introduced by Furmanski et al. (Furmanski, Azuma, and Daily 2002) and later used by Bichlmeier and Navab (Bichlmeier and Navab 2006). In those works the application had to have a model of the back faces of the cutaway to convey the limits of the technique and to aid motion parallax. Likewise, the extended X-ray tool will also need the knowledge of back faces, this time, of the occluding object itself (similar to the back faces described by Diepstraten et al. (Diepstraten, Weiskopf, and Ertl 2003)). This is typically trivial, since the model of the occluding object is necessary for both the predefined and procedural mask.

4.5.4 Discussion

We have presented a technique for enhancing the augmentation of occluded objects. The proposed technique relies on a predefined importance mask and is used during runtime to specify which fragments of the input video to preserve.

For this technique, a mask has to be created for all scene objects that will act as occluding objects; moreover, these objects need to be static or tracked. This implies the need to consider problems typically related to tracking and registration, although the technique is somewhat resistant to these issues as objects are not augmented, but information is preserved.

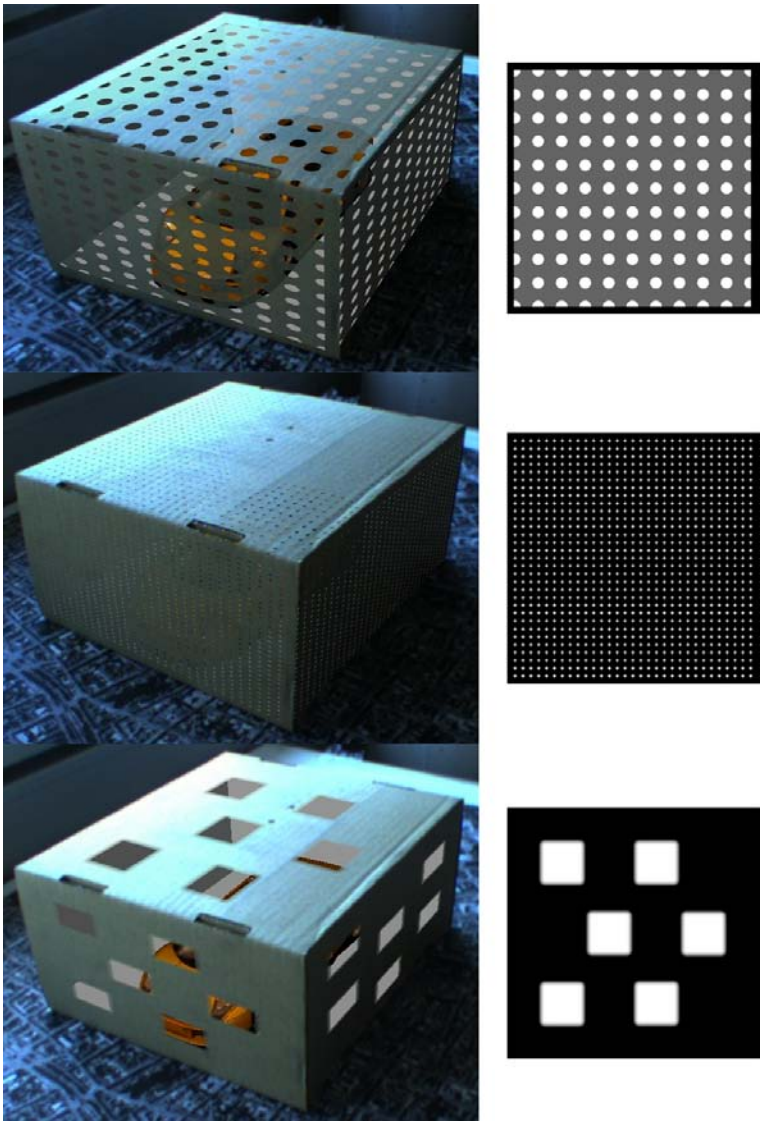


Figure 69. Mask examples. These examples explore different masks for X-ray visualization, demonstrating that the resulting effect is highly dependent on the pattern embedded in the mask. On the right are the masks used.

The presented technique allows for objects without inherent texture (structure) to still preserve information. It is also resistant to illumination view changes, since the information preserved does not depend on the current viewing conditions. Figure 69 presents different example masks for the preservation of context; notice how different the results can be between input masks. An important research direction is then to find the characteristics that masks must have in order to provide a good perception of depth.

The work presented here relies on a preprocessing step that cuts computation times drastically. Finally, the importance mask can come from any number of sources and be specifically tailored to the task (e.g., by visual artists).

4.6 Summary

We have presented a series of techniques for enhancing the perception of depth of virtual objects based on context preservation. These techniques use different props to ensure that the information of the physical occluding objects is not overwritten by the augmentations. Additionally, a small extension of the definition of X-ray vision was given.

All of the techniques presented share the same core concept: Information in the contextual part of the image should not be ignored, but should rather be exploited. Table 2 summarizes the advantages, disadvantages, and requirements of each technique. The rows of the table list all of the techniques presented in this chapter, while the columns illustrate the advantages, disadvantages and requirements of each technique.

The choice of which technique to use depends on the desired visualization and the conditions of the augmentation. For example, in the case of edge detection, better results may be obtained with images generated from rendered objects. This is because rendered objects are not affected by image noise and are therefore easier to process. However, this implies that the resulting edges are also subject to tracking errors and poor registration. Images from the video stream do not suffer from registration problems, but are subject to noise that can cause artifacts in the computed visualization. Predefined masks do not suffer from noise artifacts, but a preprocessing step is necessary. In the end, the choice of which technique to use will depend on the task. In addition, available resources, low uncertainty tracking, low powered devices, and a prior knowledge of the scene will play an important role in this decision.

We have also expanded on the definition of X-ray visualization that provides an artificial context of the occluded area. This definition empirically improves the visual results of the augmentations. Whether any of the techniques presented in this chapter effectively improve the perception of depth should be formally tested with user studies. The imagery presented throughout this chapter suggests that the answer to this is yes, but proper analyses are necessary, such as those conducted by Interrante et al. (Interrante, Fuchs, and Pizer 1996).

This chapter used a different definition of context than that used in Chapter 3. Here, we considered the contextual information to be spatially occluding information. Most of the examples consider this to come from the video feed. The next chapter will use a similar definition, where context is referred to as *spatially neighboring information*. This time contextual information will be used with a different goal: to direct *human visual attention*.

	Requires knowledge of Physical Occluding Object	Requires Tracked Virtual Occluding Model	Sensitive to Illumination Changes	Importance Mask Generation	Styling of Importance	Heavy Computation
Linked virtual models (Section 4.3)	Yes	Yes	No	Runtime	Predefined	No
Inferred information from video feed (Section 4.4)	No	No	Yes	Runtime	Based on Video	Yes
Pre defined mask (Section 4.5.1)	Yes	Yes	No	Preprocessed	Based on Video	No
Procedurally generated mask (Section 4.5.2)	Yes	Yes	No	Runtime	Based on Video	Depends on Engine

Table 2. Advantages and disadvantages. This table illustrates the main advantages and disadvantages of the techniques for occlusion management introduced in this Chapter.

Chapter 5
Using Context for Visual Attention Direction

The previous chapter introduced the notion that video information on an AR system is a resource that has to be handled carefully. By providing naïve X-ray augmentations, one can override important information that gives the augmentation context. The video itself becomes the contextual information to carefully preserve and manage. This chapter also considers portions of the video to be the contextual information, but for a different purpose: for directing attention. Previously, augmentations were designated as the *focus*, while the co-located video was designated as the *context*. Here, instead, the entirety of the video information is classified as either focus or context, and no virtual information is augmented.

Attention direction is an important tool for graphical interfaces. In mixed reality, or more precisely mediated reality, it can be used to draw the attention of a user to certain objects of a scene, be it to indicate danger, to supplement detailed information, or to guide the user to a destination. Attention direction techniques visually discriminate interesting objects (focus) from nearby related objects (context). There exist several strategies to achieve this, for example, by changing the color, adding augmentations or by distorting the desired area of attention (Kosara, Hauser, and Gresh 2003).

Not all of the strategies are universally effective, and the choice of which to use depends heavily on the focus and context objects themselves. For example, suppose we were to draw the attention of the user to a particular region by drawing a circle around it. The effectiveness of this technique depends on parameters such as the color or size of the circle. Moreover, the choice of these parameters depends on the scene, for example, by using a color that offers sufficient contrast with the rest of the image.

Consequently, an adaptive discrimination of scene objects is needed, (i.e., the attention direction strategy has to be constantly adjusted). Specifically, in mixed reality applications based on live video, one cannot easily impose constraints on visible objects or camera movements. What we suggest in this chapter is to analyze the image and compute the already existing areas of attention. Then, by modifying the image by changing its color properties, we can have the salient portions inside the desired focus region.

The work presented in this chapter has gone through a number of iterations and improvements steered by user studies. Throughout this chapter, we will present the two major revisions of our attention-direction technique, as well as the user studies that drove them. The first technique aims at reducing the contrast solely in the context region of the image, while the second provides a more advanced mechanism for the control of salient regions in the entire image. We present several application examples from the field of mediated reality.

5.1 Overview

The goal of the techniques presented in this chapter is to direct the attention of users to a region designated by the system, chiefly by manipulating the contextual information. As the chapter progresses, new constraints are added, such as temporal and spatial coherence as well as maximal perceptible changes. Either of the two versions presented in this chapter can be divided into two phases: an analysis phase and a modulation phase.

The analysis phase takes the input image to be modulated and provides a measure of the current salient regions. In our work, we use slightly modified versions of the visual salience model of Itti et al (Itti, Koch, and Niebur 1998). However, the techniques are not bound to this particular model other models such as that of Achanta et al. (Achanta et al. 2008) or Parkhurst et al. (Parkhurst, Law, and Niebur 2002) may be used.

The modulation phase uses the measurements of the image given by the analysis phase and applies a number of heuristic steps in order to shift the attention to a desired location. The techniques presented vary in complexity and efficiency. On the one hand, the technique presented in Section 5.2 is a single pass technique with disregard to spatial and temporal coherence. The technique presented in Section 5.3, on the other hand, is a heavier multi-pass technique that attempts at reducing the amount of change in the resulting image.

Although both techniques are capable of performing at interactive frame rates, only the second was targeted at video streams. The first attempt focuses on the modulation of single images given a classification mask.

5.1.1 Classification mask

Both techniques presented require that a mask is given as input. This mask should be provided as a 2D bitmap aligned with the video feed. The purpose of the mask is to provide a classification of which portions of the image are considered focus and which context. In the general case, the sources of this mask are quite diverse, and most notably are the result of tracking the focus object and generating the mask on the fly. However, due to the current limitations on tracking technologies, the masks used throughout this chapter are rotoscoped –a technique in which we matched frame by frame the video movement with a mask.

5.2 Attention direction by manipulating the HSV values of the image

The techniques presented in this chapter are based on an analysis of the original input image. They differ in the particular details of the modulation and the color space in which they work. However, they both share the concept that the image is first analyzed before modulation takes place. The first technique in this chapter for attention direction works on the Hue Saturation Value (HSV) space (Foley et al. 1995). We draw the user's attention to the desired focus object by de-emphasizing the saliency of the context, while the focus region is left unmodified. This technique uses a heuristic approach to apply a small amount of change to the image. We also report on the performance of our technique as tested by a formal user study with an eye tracker. Analysis of participants' gaze shows that images processed with our technique can draw participants' attention to a focus region significantly faster and can retain it for a significantly longer time than the original unmodified images.

5.2.1 Analysis of attention areas

In order to direct the attention of the viewer to a particular location, one can either increase the saliency of that location (focus), decrease the saliency of its surroundings (context), or do both. Because the saliency of a location is a combination of several conspicuities, the final goal is to modulate the appropriate conspicuities by location.

Figure 70 illustrates the calculated conspicuities for lightness, saturation and the opponent colors. Every conspicuity map shows a plotted scanline on the side for comparison. The topmost image is the input. The image on the second row is the lightness conspicuity. For purposes of illustration, we show positive values in green and negative in red (i.e., dark objects near light ones have negative conspicuity). The image on the third row shows the saturation conspicuity, again with positive values in green and negative in red. The image on the fourth row shows in red the Red-Green conspicuity and in green the Blue-Yellow conspicuity. Finally, the image on the bottom row is the total saliency map of the image.

Once the conspicuities have been calculated, we compute the total saliency of the image. Itti and Koch (Itti and Koch 1999) addressed the problem of combining conspicuities with four distinct strategies: (a) simple normalized summation, (b) linear combination with learned weights, (c) global non-linear normalization followed by a summation, and (d) local non-linear competition between salient locations. Although strategies (c) and (d) performed the best, they are computationally prohibitive for interactive frame rates. Strategy (b), denoted as $N(.)$, yields satisfactory results, but is still computationally expensive. Lee et al. (Lee, Kim, and Choi 2007) devised a simplified faster implementation of $N(.)$ that runs in real time on the GPU, which we have adopted.

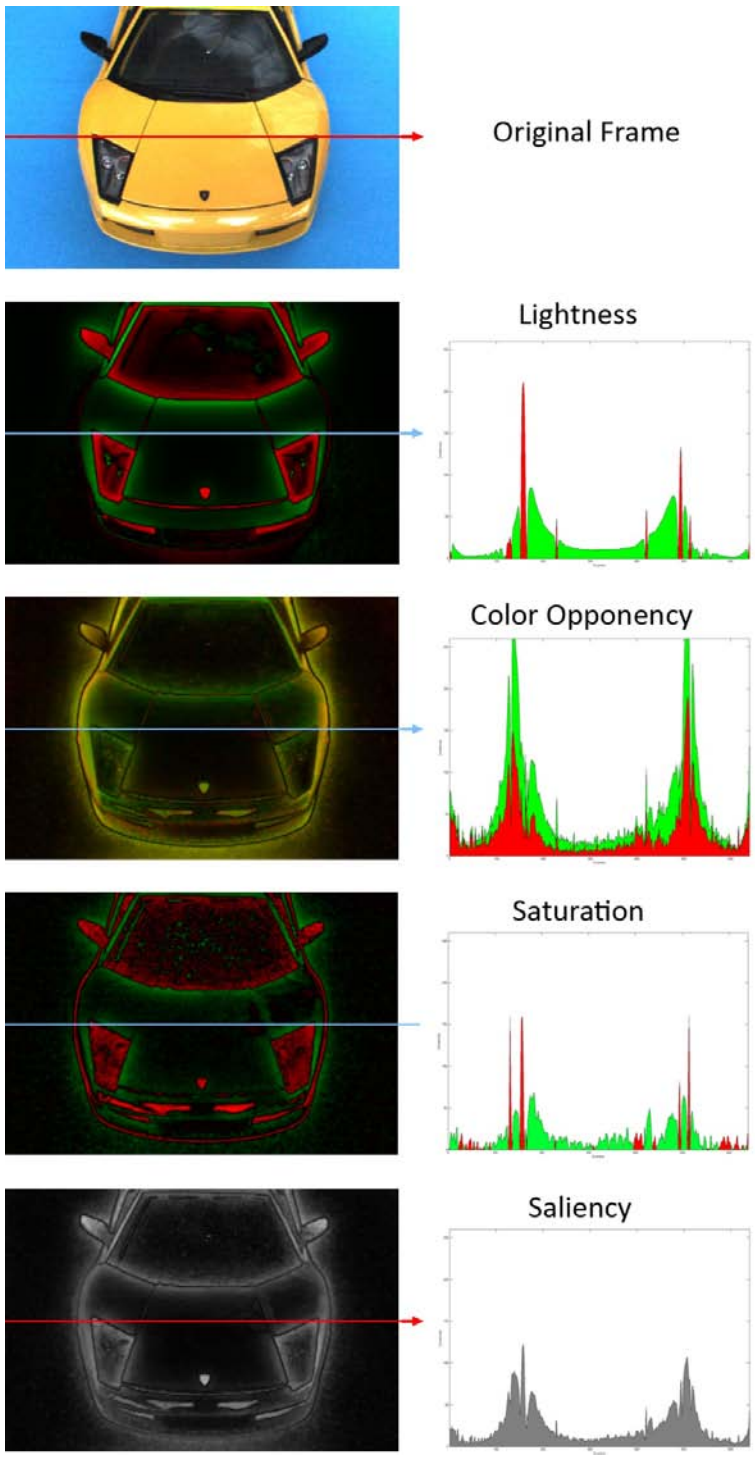


Figure 70. Conspicuities. A set of conspicuities obtained from a frame of the input video along the total saliency map. For comparison, every image shows a plotted scanline.

5.2.1.1 Saliency computation implementation

Video analysis provides a normalized quantitative measure of the current saliency of the image in focus and context areas. Figure 71 shows a diagram of the following steps:

- Feature Extraction
- Pyramid Generation
- Conspicuities Computation
- Normalization
- Saliency Computation

We compute these as follows.

Feature Extraction

First, we calculate the image feature maps in the dimensions lightness, saturation, Red-Green opponents and Blue-Yellow opponents.

Lightness. This is the fragment's Value component in the HSV space.

Saturation. This is the fragment's Saturation component in the HSV space.

Opponent Colors. Itti et al. (Itti, Koch, and Niebur 1998) provide formulas for calculating the opponent color values. We use these formulas summarized in the following:

Let r, g and b be the RGB values of the current fragment. The red-green opponent, O_r, and blue-yellow opponent, O_b, are then,

$$O_r = \left(r - \frac{g+b}{2}\right) - \left(g - \frac{r+b}{2}\right),$$

$$O_b = \left(b - \frac{r+g}{2}\right) - \left(\frac{(r+g-|r-g|)}{2} - b\right).$$

We compute all these quantities simultaneously in a single fragment shader and encode the results in a 32-bit floating point texture. Lightness (L), saturation (S), red-green nents(O_r), and blue-yellow opponents (O_b) are passed, respectively, in the R, G, B and A channels of the texture.

Pyramid Generation

We calculate a six-level mip-map pyramid (Williams 1983) from the feature textures, effectively creating 24 feature maps as described by Itti et al. (Itti, Koch, and Niebur 1998), six each for lightness, saturation, Red-Green color-opponents, and Blue-Yellow color-opponents. This is done in six passes and the result is delivered in another floating point texture. Lee et al. (Lee, Kim, and Choi 2007) used the built-in graphics support for this mip-map computa-

tion, but we require floating point non-square textures, which have no built in hardware support for mip-mapping. Therefore, we compute the image pyramid in multiple render passes using a custom fragment shader.

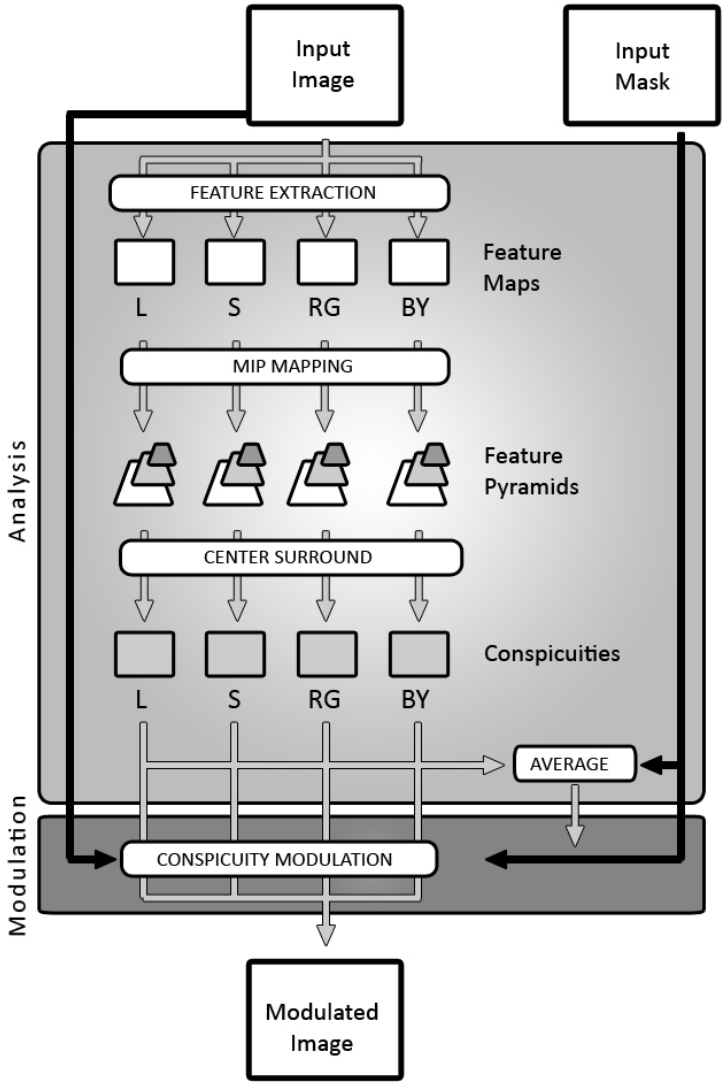


Figure 71. Algorithm overview. The Analysis stage is similar to that described by Itti et al. (Itti, Koch, and Niebur 1998), except for the computation of the average of the values of the focus. Modulation considers conspicuities, averages, image input and mask to produce the final result.

Conspicuities Computation

The next step computes the conspicuities for each separate dimension. This is done by calculating the differences across pyramid levels. The input is all 24 feature maps and the output is four conspicuity maps. The difference across pyramid levels is calculated with the center-surround technique. This technique calculates the relation of a location to its surroundings by checking the difference across fine and coarse levels of the previously computed pyramid.

The following formula calculates the center-surround of a location without the $N(.)$ operator (Itti, Koch, and Niebur 1998):

Let k_i be the fragment's feature k on pyramid level i. The conspicuity c_k is then defined as:

$$c_k = \frac{\sum_{n=2}^{n=2} \sum_{m=3}^{m=4} k_n - k_{n+m}}{d},$$

where $k \epsilon \{L, S, O_r, O_b\}$ and d=4, the number of dimensions.

There is an important difference relative to previous work: We do not add up the two color-opponent conspicuities in this stage because we need them separately for the saliency modulation stage.

Normalization

Due to the expensive computation of Itti's $N(.)$ operator, we instead use a normalization that considers the global conspicuity maxima, as described by Lee et al. (Lee, Kim, and Choi 2007). This has the effect of reducing non-contributing high-frequency artifacts, such as those present in the color-opponents map of Figure 70. The normalized conspicuity is defined as follows:

Let $max(c_k)$ be the maximum conspicuity value of the feature k of the whole image. The normalized conspicuity at every location $\hat{c}_k$ is then

$$\hat{c}_k = \frac{c_k}{\max(c_k)},$$

where $k \epsilon \{L, S, O_r, O_b\}$.

Saliency Computation

Saliency is a linear combination of the normalized conspicuity maps. We use saliency throughout this section for illustration purposes and to validate our results, but not for the modulation step. The computation of the saliency S at a given location is

$$S = \frac{\sum \hat{c}_k}{\mathrm{d}},$$

where $k \epsilon \{L, S, O_r, O_b\}$ and d=4, the number of dimensions.

Conspicuities, as defined by Itti et al., are positive floating point values. They indicate the difference between an object and its surroundings. This poses a certain problem when trying to modulate them. For example, a high lightness conspicuity does not tell us whether the current location is a dark location on light surroundings or vice versa. In Itti's work (Itti, Koch, and Niebur 1998), the conspicuity is calculated by the sum of absolute differences

across mip-map levels. This would imply a loss of the conspicuity sign, which tells us whether the current location is, for example, dark on a light surrounding or vice versa. Therefore, we store the sign before calculating the absolute value and add it again before passing it to the normalization step. This guarantees that in the modulation step we know whether to darken or lighten the current fragment. Figure 70 shows a sequence of conspicuities, illustrating their sign.

5.2.1.2 Modulation threshold

The vector of conspicuities tells us how different every location is relative to its surroundings. But, in order to modulate any conspicuity, we need to have a threshold that tells us whether any changes are necessary. Since we modulate several dimensions, this threshold is actually a collection of values. The threshold values will be referred as t_K, where k is the given dimension throughout this chapter. These values should come from the focus area, since we are interested in drawing attention to it. However, in the general case, the focus region is itself a collection of locations. The vector t_K can be, for example, the minimum or the maximum conspicuity values. In our work, we have obtained satisfactory results by using the average conspicuities inside the focus region. This threshold can be thought of as the strength to which our modulation will be applied and the application programmer can allow the user to control it.

5.2.2 Modulation of attention areas

We now have the components necessary for the modulation step: the vector of conspicuities at every location, and the average conspicuities in the focus area. We proceed to modify the video image only in the context region, while the focus remains unchanged. We modulate all conspicuities in the same dimension, where they were measured: lightness, saturation, and color-opponents (hue).

A naïve method would heavily decrease all the conspicuities of the context area regardless of what is present in the scene (e.g., see Figure 72). This increases the emphasis of the focus object, but at the same time drastically reduces the contribution of the context, since all pixels are modified, whether the change is necessary or not. Better discrimination can be achieved by choosing an appropriate dimension (saturation or lightness, for example) on a per-pixel basis in which to modify the image. Our saliency analysis identifies the dimensions in which the saliency of a location is greater than the average saliency of the focus. Each of these dimensions is then modulated in those locations where the current conspicuity is greater than the average conspicuity of the focus (defined by t_k). This means that modulations are not applied equally to all fragments. For example, some might need a strong saturation modulation, but no lightness changes. Modulation is performed in three sequential steps: lightness modulation, saturation modulation, and color-opponent modulation.

Figure 72. Naïve technique. Isolating an object is possible by heavily modifying saturation and lightness. This, however, reduces the contributions of the context to the overall scene.

5.2.2.1 Lightness modulation

Modulation of lightness is performed by changing the fragment's color space from RGB to HSV space. Once this conversion is done, we compute the difference between the fragments' normalized lightness conspicuity ($\hat{c}_L$) and the average lightness conspicuity of the focus (from t_L as defined in Section 5.2.1.2). We increase or decrease the lightness (L) according to this value.

5.2.2.2 Saturation modulation

Modulation of saturation is performed by changing the fragment's color space from RGB to HSV space. Once this conversion is done, we compute the difference between the fragments' normalized saturation conspicuity ($\hat{c}_S$) and the average saturation conspicuity of the focus (from t_S as defined in Section 5.2.1.2). We increase or decrease the saturation (S) according to this value.

5.2.2.3 Color-opponents modulation

This modulation also takes place in the HSV space, this time on the hue channel; however, its computation is a bit more complex. The opponent process theory states that the color channel pairs red-green and blue-yellow are each mutually opposing. The HSV space arranges colors in a cone form where the hue is encoded by the angular position around the cone (Foley et al. 1995). The red color is at 0°, yellow at 60°, green at 120°, cyan at 180°, blue at 240° and magenta at 300° degrees (see Figure 73). Having this knowledge, we can decrease the color-opponent conspicuity of an object with Algorithm 6.

```
Let H_i be the hue of fragment i
Let RG_i be the red-green conspicuity of i
Let BY_i be the blue-yellow conspicuity of i
Let A_f^rg be the average red-green conspicuity of the focus
Let A_f^by be the average blue-yellow conspicuity of the focus

If ((RG_i - A_f^rg) > (BY_i - A_f^b)) then
      θ_b = Distance of H_i to the blue hue
      θ_y = Distance of H_i to the yellow hue
      If (θ_b < θ_y) then
            Move H_i towards blue by BY_i - A_f^by value
      Else
            Move H_i towards yellow by BY_i - A_f^by value
      End if
Else
      θ_r = Distance of H_i to red hue
      θ_g = Distance of H_i to green hue
      If (θ_r < θ_g) then
            Move H_i towards red by RG_i - A_f^rg value
      Else
            Move H_i towards green by RG_i - A_f^rg value
      End if
End if
```

Algorithm 6. Modulation of color-opponents on HSV space. This algorithm modulates the color-opponents of a pixel to decrease the conspicuity of the opponents.

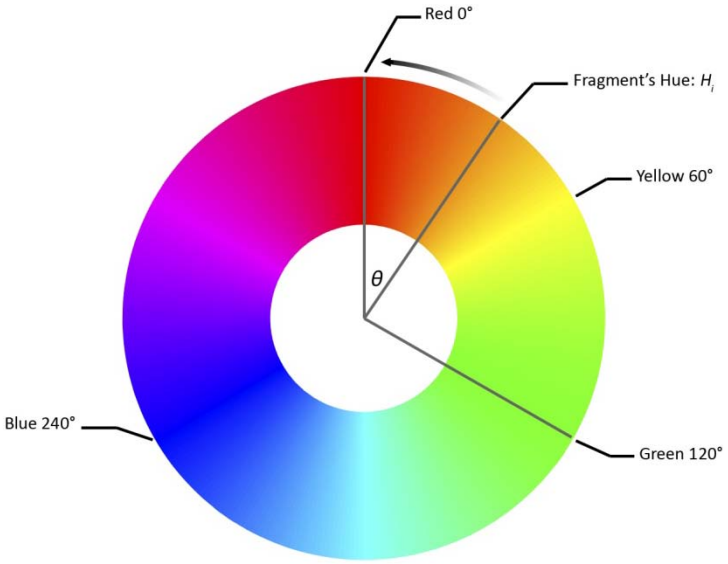


Figure 73. Color-opponents modulation. We modulate color-opponents by moving the fragment's Hue, H_i, away from the color-opponents given the angular distance θ. This example checks the blue-yellow opponency and illustrates how H_i is moved towards red (away from yellow).

Figure 73 illustrates the modulation process. The fragment's hue, H_i, does not tell us anything about its surroundings. Such information is encoded in the conspicuities. For example, assume that the fragment has stronger blue-yellow opponents value than red-green opponents. This means that blue and yellow give a high contrast to the fragment and we should move its hue away from them. We calculate then the distance of the fragment's hue to both red and green. The distance to red (θ) is shorter and as a result the hue is moved towards it. There is, however, an important detail that must be taken into account: In the opponent process, the red color is equidistant to both yellow and blue, but in the hue wheel, red is closer to yellow than to blue. This can be easily solved by weighting down the distance to blue.

5.2.3 Validation

To test the performance of our saliency modulation technique, we carried out a formal user study with an eye tracker. The goal of the study is to test whether our technique can direct the visual attention of the user to the regions of the images that we designated as focus, regardless of the information in the original image. Our hypotheses are:

- **H1.** The time before the first fixation on the focus region will be smaller for the images modulated with our procedure than for the original unmodified images.
- **H2.** The fixation time (i.e., sum of durations of all fixations on the focus) will be higher for the images modulated with our procedure than for the original unmodified images.
- **H3.** The percentage of participants that have at least one fixation on the focus region will be higher for the images modulated with our procedure than for the original unmodified images.

5.2.3.1 Experiment description

The experiment is composed of two phases: artificial and natural. The artificial set is created in order to check the effectiveness of each of the three dimension modulations (lightness, saturation and color-opponents) separately. The natural set involved images from real scenarios and is the main target of our experiment.

Artificial images

The artificial images were created with a popular graphics package, Adobe Photoshop. The images included an arrangement of stimuli in all of the three dimensions for a total of three image sets: artificial-lightness, artificial-saturation, artificial-color-opponents. Each image is created with a single focus area and is subsequently modulated with a single step of our technique (lightness modulation, saturation modulation, or color-opponents modulation) to change the focus to a different location. The top image pair of Figure 74, for example, shows

an unmodified image of the artificial-lightness set (left column), where the focus is the object with lower lightness, and the same image after our modulation technique (right column). This is the result of our technique that tries to diminish the lightness conspicuity of the image, as described in Section 5.2.2.1. The middle image pair of Figure 74 shows a pair of images of the artificial-saturation set. This is the result of our technique that tries to diminish the saturation conspicuity of the image (see Section 5.2.2.2).

The effect of the modulation on color-opponents of the four colors, red, green, blue and yellow was tested by, for example, an image with one of these colors as the background and three circles with the remaining colors. The bottom image pair of Figure 74 shows an example image from the artificial-color-opponents set for yellow. According to the opponent process theory, the blue circle is the most salient of the three circles. The right image shows the result after saliency modulation where the attention of the subject is directed to the red circle. Notice how the blue circle is moved towards the cyan hue. This is the result of our technique, which tries to diminish the high blue-yellow conspicuity by moving the values around the hue wheel, as described in Section 5.2.2.3.

Natural images

The natural image set is a collection of photographs of outdoor environments. These images were selected to include gardens, streets, people gatherings, city landscapes, and so on. They included many other dimensions of attention such as shape, orientation or texture detail which are not considered for modulation by our technique. The designated focus regions (i.e., where we wanted to direct the attention of participants) are scattered around, including and excluding human faces, perspective vanishing points on the horizon, etc. These focus regions are always placed away from the region of the original image with the highest salient. Every image is modulated with all three dimensions sequentially: lightness, saturation, and color-opponents.

Figure 75 and Figure 76 each show an image before and after our modulation procedure. Figure 75 shows a bridge over a gorge with a wall of rocks in back. The modulated image tries to direct the attention of the user towards a rock on the upper part of the wall. Figure 76 shows a busy street with multiple colors, perspective lines, faces and so on. The modulated image tries to direct the attention towards the second farthest lamp on the right hand side. The order in which the two sets of images were presented was randomized, as was the order of presentation of images within each set. There were a total of 24 images in the artificial set (6 for artificial-lightness, 6 for artificial-saturation and 12 for artificial-color-opponents) and 27 in the natural set. Each image was shown for 5000ms. Between images, a blank slide with a cross in the middle was shown for 2000ms in order to standardize the participant's initial gaze position before the image was presented.

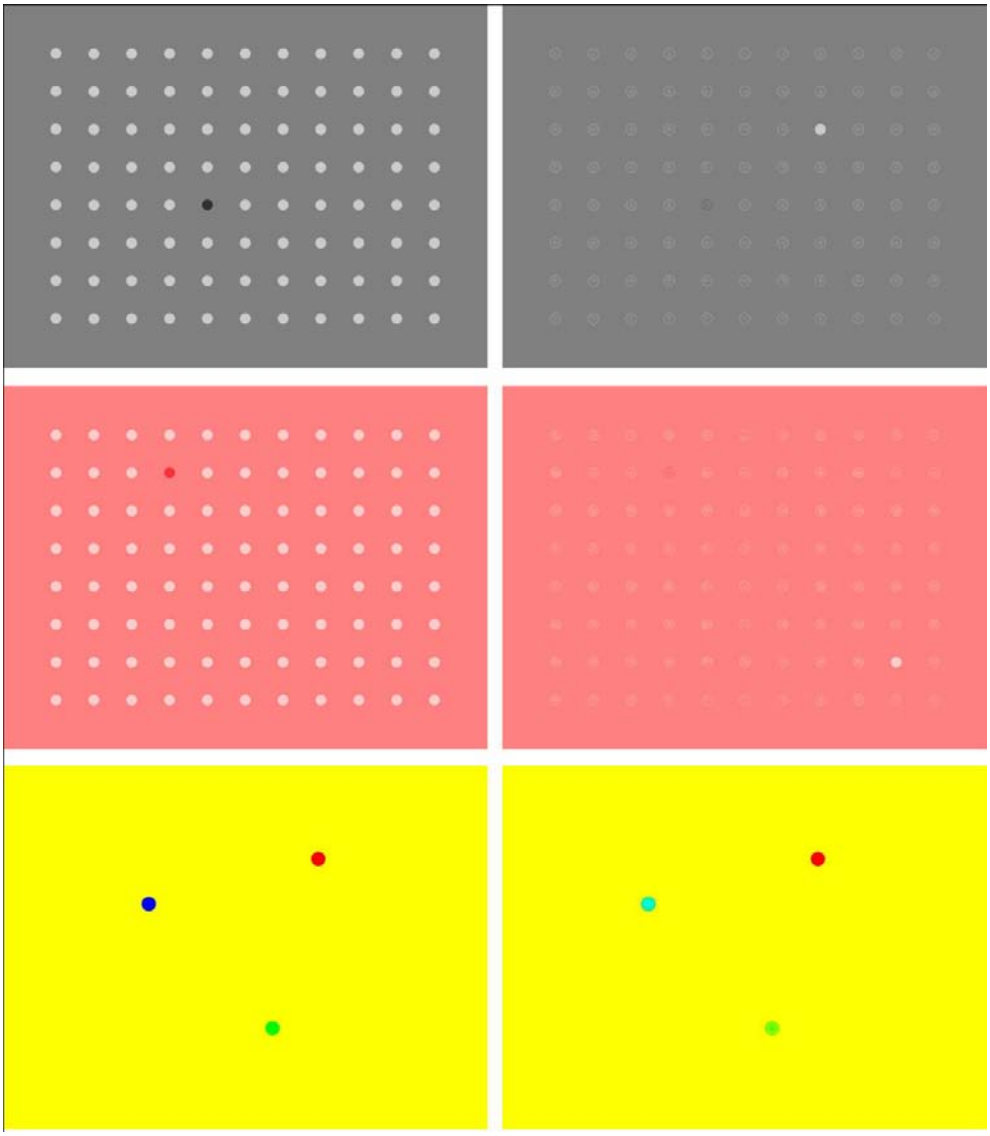


Figure 74. Example images from the artificial image set. Images were created artificially, with one "natural" salient object. (Left column) Original unmodified images. (Right column) Modified images.

Figure 75. Modulation of bottom-up salient features of context. (Top) Original image. (Bottom) Result of our modulation technique. The saliency of pixels outside the focus region is automatically decreased. Pixel values of the modulated image differ on average by 2.25% from their counterparts in the original image.

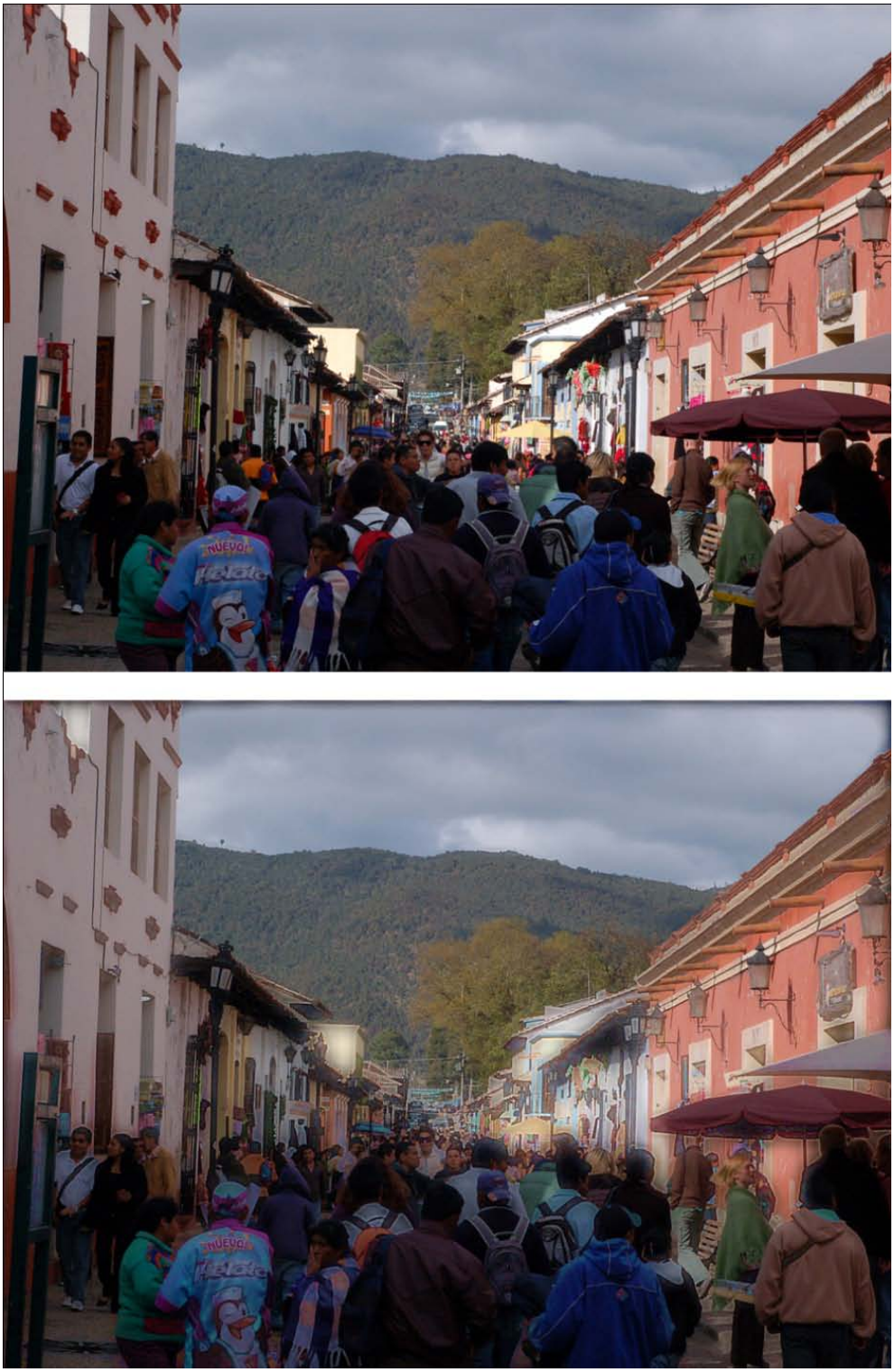


Figure 76. Picture of a busy street before and after modulation. (Top) Original image. (Bottom) Modified image. Attention was directed towards the second closest lamp on the wall on the right. The pixel values of the modified image differ on average by 1.84% from their counter-parts on the original image.

Participants

The test was performed on 30 participants (6 female, 24 male) between 24 and 34 years old ($\bar{x}$=28). All participants had no known color sensitivity deficiencies; which was confirmed by an Ishihara color test. Participants were compensated with a gift certificate from a popular online shop.

Apparatus

The tracking device was an SMI desktop-mounted eye tracker, operating at 60 Hz. The stimuli were presented in the form of a slide show on a 19" monitor at a 70cm distance from the participant. The resolution of the images was 1280×960, and all were presented without resizing in order to avoid interpolation by the graphics unit.

5.2.3.2 Analysis

Analysis is performed with paired-samples t-tests and levels are adjusted (Bonferroni) to ensure a significant level of 5% (p<0.00625). We discarded the data for the first 200ms after the image was shown, based on the assumption that this is the amount of time the brain needs to build the saliency map (Itti and Koch 2000).

Lightness	Original [ms]	Modulated [ms]	Speed up [ms]	Significance
H1	4069 (σ=766.9)	2141.4 (σ =1079.5)	−1927.6	p<0.001

Table 3. Results for hypothesis H1 for the artificial-lightness set. This table shows the average time before participants fixated on the focus.

Lightness	Original [ms]	Modulated [ms]	Extra Fixation [ms]	Significance
H2	102.61 (σ =92.51)	819.73 (σ =402.2)	717.12	p<0.001

Table 4. Results for hypothesis H2 for the artificial-lightness set. This table shows the sum of the total fixation time that participants spent on the focus.

Lightness	Original [%]	Modulated [%]	Difference [%]
% participants with fixation by 500ms	0	33.33	33.33
% participants with fixation by 1000ms	5.83	50	44.17
% participants with fixation by 5000ms	20	79.16	59.16

Table 5. Results for hypothesis H3 of the artificial-lightness set. This table shows the percentage of participants that had at least one fixation on the focus region by the first 500ms, 1000ms, and 5000ms.

Analysis of the artificial images set

Table 3 to Table 11 show the results of the tests for the artificial images of all three tested dimensions separately. Standard deviations and statistical significances are shown. It is important to mention that there is no interaction among the sets, since the images used for each set are different.

Saturation	Original [ms]	Modulated [ms]	Speed up [ms]	Significance
H1	4596.1 (σ =475.2)	1736 (σ =867.2)	−2860.1	p<0.001

Table 6. Results for hypothesis H1 for the artificial-saturation set. This table shows the average time before participants fixated on the focus.

Saturation	Original [ms]	Modulated [ms]	Extra Fixation [ms]	Significance
H2	35.48 (σ =43.01)	742.70 (σ =329.33)	707.21	p<0.001

Table 7. Results for hypothesis H2 for the artificial-saturation set. This table shows the sum of the total fixation time that participants spent on the focus.

Saturation	Original [%]	Modulated [%]	Difference [%]
% participants with fixation by 500ms	0	38.33	38.33
% participants with fixation by 1000ms	1.66	60	58.34
% participants with fixation by 5000ms	17.5	75	57.5

Table 8. Results for hypothesis H3 for the artificial-saturation set. This table shows the percentage of participants that had at least one fixation on the focus region by the first 500ms, 1000ms, and 5000ms.

Artificial-Lightness. Table 3 to Table 5 show the results of the analysis of the artificial-lightness set. Participants had a first fixation on the focus significantly faster on the images modulated with our technique (2141.4ms) than on the original unmodified images (4069ms) (Table 3). Participants also spent significantly longer time fixated on the focus of the images modulated with our technique (819.73ms) than on the original unmodified images (102.61ms) (Table 4). By the first 500ms, 33.33% of the participants had had already at least one fixation on the focus of the modulated images, while none of the participants had fixated on the focus of any of the original unmodified images. By the end of the stimulus, 59.16% more of the participants had at least one fixation on the focus on the modulated images than on the original unmodified images (Table 5).

Artificial-Saturation. Table 6 to Table 8 show the results of the analysis of the artificial-saturation set. Participants had a first fixation on the focus significantly faster on the images modulated with our technique (1736ms) than on the original unmodified images (4596.1ms) (Table 6). Participants also spent significantly longer time fixated on the focus of the images modulated with our technique (742.7ms) than on the original unmodified images (35.48ms) (Table 7). By the first 500ms, 38.33% of the participants had had already at least one fixation on the focus of the modulated image, while none of the participants had a fixation on the focus of any of the original unmodified images. By the end of the stimulus, 57.5% more of the participants had at least one fixation on the focus of a modulated image than on its original unmodified image (Table 8).

Color-opponents	Original [ms]	Modulated [ms]	Speed up [ms]	Significance
H1	1896.6 (σ= 713.4)	2077.3 (σ =632.6)	180.7	p=0.259

Table 9. Results for hypothesis H1 for the artificial-color-opponents set. This table shows the average time before participants fixated on the focus.

Color-opponents	Original [ms]	Modulated [ms]	Extra Fixation [ms]	Significance
H2	848.85 (σ =355.2)	833.33 (σ =279.9)	−15.52	p=0.840

Table 10. Results for hypothesis H2 for the artificial-color-opponents set. This table shows the sum of the total fixation time that participants spent on the focus.

Color-opponents	Original [%]	Modulated [%]	Difference [%]
% participants with fixation by 500ms	17.5	22.91	5.41
% participants with fixation by 1000ms	47.08	47.5	0.42
% participants with fixation by 5000ms	82.08	79.58	−2.5

Table 11. Results for hypothesis H3 for the artificial-color-opponents set. This table shows the percentage of participants that had at least one fixation on the focus region by the first 500ms, 1000ms, and 5000ms.

Artificial-Color-Opponents. Table 9 to Table 11 show the analysis results for the artificial-color-opponents set. The results for hypotheses H1 and H2 are not found to be statistically significant. Participants tended to spend less time before fixating for the first time on the focus on the original unmodified images (1896.6ms), than on the modulated counterpart of the same image (2077.3ms)(Table 9). Participants also tended to spend more time fixated on the focus on the original unmodified images (848.85ms), than on the modulated counterpart

of the same image (833.33ms) (Table 10). Interestingly, for the third hypothesis (H3), by the first 500ms, the modulated images of the color-opponents dimension called the attention of an extra 5.41% of the participants, but by the end of the presentation of the stimulus, the modulation had a negative impact of 2.5% (Table 11).

As can be seen, lightness and saturation are successfully modulated, confirming all three hypotheses. In average, lightness and saturation modulation all images presented a significant speed up of first fixation time and a significant increase on total fixation time, as well as a higher percentage of participants having at least one fixation on the focus region. However, no statistically significant results are obtained for modulation of the color-opponents dimension.

Analysis of the natural images set

Table 12 to Table 14 show the results of the tests for the natural images. The results are placed in contrast with the hypotheses formulated earlier. Standard deviations and statistical significances are shown. It is important to note, that each image in this set is modulated across all three dimensions (unlike the images in the artificial sets, which were each modulated in only one dimension).

Natural	Original [ms]	Modulated [ms]	Speed up [ms]	Significance
H1	3842.3 (σ =412.3)	3382.8 (σ =463.2)	−459.5	p<0.001

Table 12. Results for hypothesis H1 for the natural set. This table shows the average time before participants fixated on the focus.

Natural	Original [ms]	Modulated [ms]	Extra Fixation [ms]	Significance
H2	176 (σ =65.6)	338.3 (σ =115.6)	162.3	p<0.001

Table 13. Results for hypothesis H2 for the natural set. This table shows the sum of the total fixation time that participants spent on the focus.

Natural	Original [%]	Modulated [%]	Difference [%]
% participants with fixation by 500ms	2.9	8.3	5.4
% participants with fixation by 1000ms	11.2	20.4	9.1
% participants with fixation by 5000ms	40.9	55.5	14.6

Table 14. Results for hypothesis H3 for the natural set. This table shows the percentage of participants that had at least one fixation on the focus region by the first 500ms, 1000ms, and 5000ms.

H1. Duration before first fixation. Table 12 shows the average time that passed before the participants fixated on the focus region for the first time in both original and modulated conditions, as well as the speed up (difference) across all natural images. Participants had a first fixation on the focus significantly faster on the images modulated with our technique (3382.8ms) than on the original unmodified images (3842.3ms).

H2. Total fixation time. Table 13 shows the average of the sum of the total time that the participants spent fixated on the focus region in the original and modulated conditions, as well as the difference. Participants spent significantly longer time fixated on the focus of the images modulated with our technique (338.3ms) than on the original unmodified images (176ms).

H3. Percentage of participants with at least one fixation. Table 14 shows the percentage of the participants that had at least one fixation on the focus region by the first 500ms, 1000ms and by the end of the stimulus, 5000ms across all natural images. By the first 500ms, 8.3% had had already at least one fixation on the focus of the modulated image, while only 2.9% of the participants had a fixation on the focus on the original unmodified counterpart, an improvement of almost double. By the end of the stimulus, an extra 14.6% of the participants had had at least one fixation on the focus on the modulated images than on the original unmodified images. As can be seen from the results, our modulation procedure can effectively draw the attention of the participants to the focus region. On average, the pixels of the modified images differ only by 1.86% from their counterparts in the original images on the natural set.

5.2.4 Discussion

All of the images shown in this section were computed at 1280×960 on a 3.0 GHz Intel Dual Core CPU with an NVIDIA GTX280 graphics card. All images were computed at a minimum of 30 fps. We used GLSL (Rost 2004) for our fragment shaders and frame buffer objects for texture handling.

Figure 77 shows graphs of the saliency map of Figure 75. The top graph is obtained from the original image; a light green overlay indicates the region we wish to set as focus. As it can be seen, the image has many high salient features competing for the user's attention. The bottom graph shows the result after our modulation process and how our technique effectively keeps the object of interest highly salient in the scene.

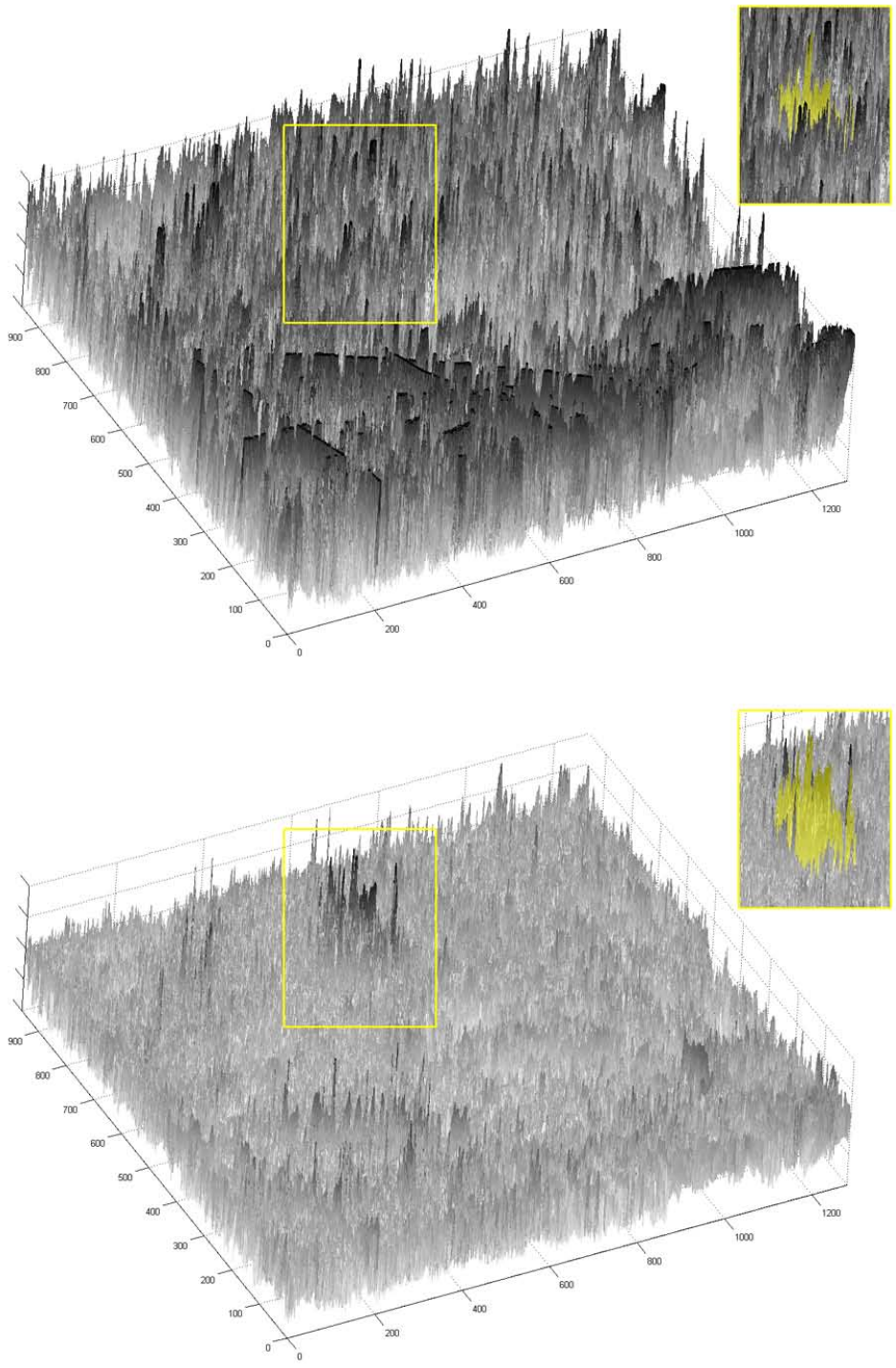


Figure 77. Comparison of saliency maps. Graphs of the saliencies of Figure 75. (Top) Original unmodified image; many locations in the scene are competing for the user's attention. (Bottom) Modulated image. Our technique eliminates competitors and clearly isolates the focus. Plots were created in an inverse scale with black as the highest value for better readability. A yellow outline denotes the position of the focus region.

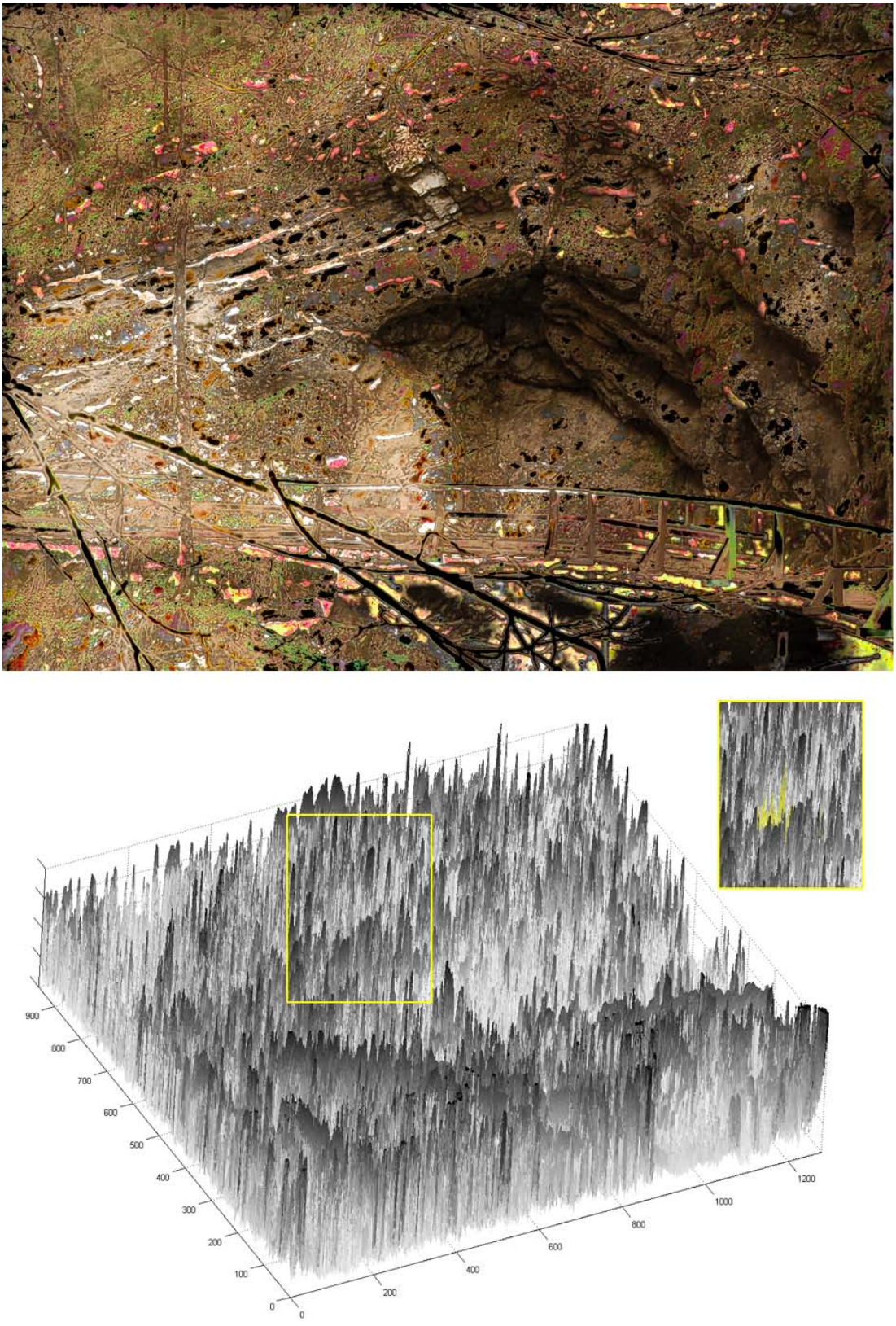


Figure 78. An erroneous weighting of the strength of the modulation and the resulting saliency map. Average pixel difference is 5.62%.

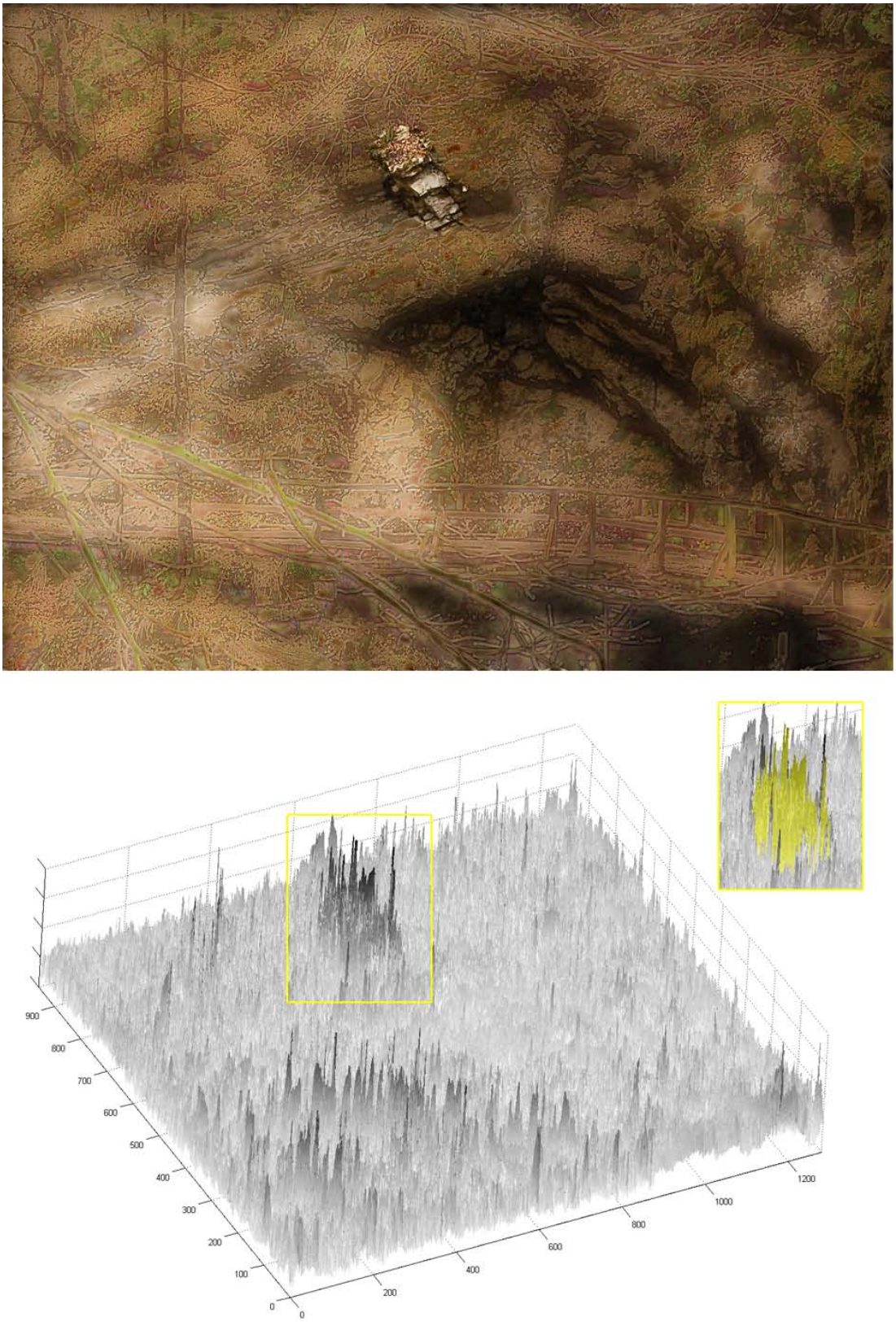


Figure 79. A better approach to weighting the modulation. This image was created by setting the threshold to half the average conspicuity values inside the focus region. Average pixel difference is 4.11%.

To find out the how much our technique changes the image we computed the average pixel difference between the original image and after our modulation procedure. This is done by calculating the square root of the sum of squared differences in the RGB space divided by the number of pixels in the image. For example, the pixels of the top image of Figure 75 differ on average by 2.25% from their counterparts in the bottom image. The total average pixel difference across all images in the user study between modified and original images is 1.86%.

In the technique presented in this section, we modulate the saliency (the set of conspicuities) by changing, the saturation, lightness and hue of the image. The amount of this change is given by the difference between the focus and context conspicuities themselves. One might be tempted to force this amount to a lesser or higher value, but the result of this tends to be unpleasant, as illustrated in Figure 78. This figure shows an image after the strength of the modulation has been increased by 500%, and the accompanying graph below shows the resulting saliency map. Increasing the strength by 500% results in uncontrolled saliencies where the focus is no longer distinguishable.

Instead of weighting up the changes to the image, a better approach is to increase or decrease the thresholds (t_k), as suggested in Section 5.2.1.2. In the work described in this section, we experienced good results by setting the threshold to the average conspicuity values of the focus region, but this can also be set by the user dynamically. Figure 79 shows an example resulting image in which the thresholds t_k have been set to half the average conspicuity values of the focus region. The isolation of the focus region is higher than that of Figure 75. However, the average pixel difference compared to Figure 75 is also higher.

In the technique presented in this section, modulation of saliency is done on the location level (per fragment). This, however, is not the only possibility; for example, one can apply the same modulation to the whole context region of the image. This might, however, be conflicting across different locations in the context area itself. For example, one location might need to increase its lightness to reduce its conspicuity, while the opposite might be true for another location. Another option would be to track a set of specific objects in the scene to which we should decrease the saliency. In this case, these tracked objects would limit the number of objects that can appear in the image. One can also apply more advanced segmentation techniques to the image but this complexity increases heavily in an uncontrolled environment. The problem also remains that since the same modulation is applied equally to a collection of fragments, these might nullify each other. Itti et al. (Itti, Koch, and Niebur 1998) use a winner-take-all algorithm that detects regions by their combined saliency rather than using individual peaks. The winner-take-all algorithm essentially detects clusters of salients, but it does not segment real objects.

To increase the saliency of the focus region, the modulation can take place in either the focus or the context. We may apply modulations in order to increase the saliency of the focus or we can decrease the saliency of the context, or both. Either of these options targets the increased overall saliency of the focus, although their visual results might differ. Throughout this section, we limit the modulation changes to only decreasing the saliency of the context.

Decreasing the saliency of a location means smoothing out the locations' differences with its surroundings, while increasing it means emphasizing differences. This emphasis effectively creates high gradients in the image with potentially unpleasant results. However, emphasis of the focus region might be necessary in cases where the focus has little initial structure. Section 5.3 presents an evolution of our modulation technique in which the modulation of the focus region is also allowed.

In this work, we apply the conspicuity modulations sequentially on a single fragment shader. However, after each change, there is no check whether the amount of modulation is sufficient and the procedure should exit (Section 5.3 addresses this problem). In this respect, the conspicuity modulation done in this section can be seen as energy functions that have to be minimized. Pock et al. have successfully shown how to compute variational methods on graphics hardware (Pock, Grabner, and Bischof 2007). Applying these methods to our work is an avenue for future work.

This section presented the first technique for attention direction. It is based on the modulation of bottom up salient features of the context region. The technique was validated with a user study confirming that it is capable of directing the attention of users to the desired focus region. One disadvantage, however, is that the images are noticeably damaged with our technique. A refinement step should focus on providing a less harmful modulation procedure. This is then what we address in the next section with the introduction of an iterative modulation process.

5.3 Attention direction by manipulating the CIEL*a*b* values of the image

The second technique in this chapter is an evolution of the one presented in Section 5.2. Besides attempting to direct the user's attention, the goal of this technique is to guide, rather than force, the attention of the user towards a specific location. The technique's aim is to apply only minimal changes to an image, while achieving a desired difference of saliency between focus and context regions of the image. This technique exhibits temporal and spatial coherence and runs at interactive frame rates.

Similar to the previous technique, all computations are carried out with GPU shaders at interactive frame rates. We present several application examples from the field of mixed reality, including directing the attention of the user to an object in a search task, and highlighting a possibly dangerous object during car maintenance.

5.3.1 Analysis of attention areas

We modulate the image on a frame-by-frame basis, in order to reflect the latest information available from a video feed. As detailed in Section 5.1, achieving this demands two general steps depending on the input before composing the final image:

- *Analysis.* During this step, we compute the conspicuities of the whole image to have a measure of the naturally salient objects in the scene.
- *Modulation.* Once we have quantified the image's conspicuities, we select and apply the appropriate modulations to the input image. The modulations are done sequentially for each of the conspicuities at multiple levels of coarseness, and ultimately produce an image whose highest salient is in the focus area.

The saliency of a location is given by a combination of its conspicuities; the final goal is then to modulate said conspicuities. We now present how the saliency of the image is analyzed so that modulation can take place. Similar to Section 5.2, for simplicity we use a few short terms for commonly used properties: lightness is simply referred as L, red-green color-opponents as O_r and blue-yellow color-opponents as O_b.

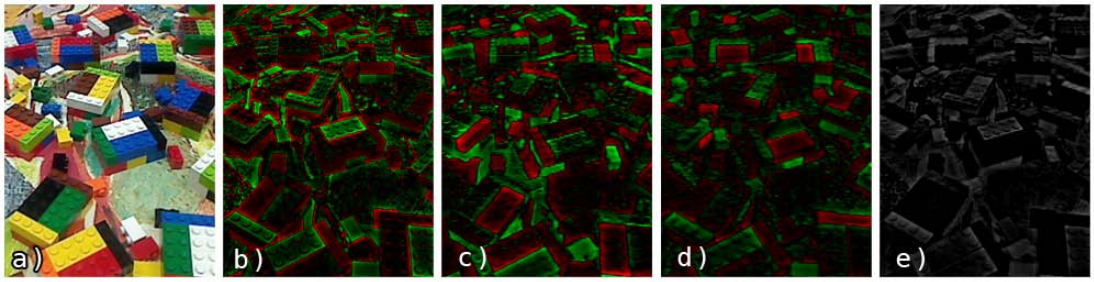


Figure 80. Illustration of conspicuities. These images illustrate the conspicuities of the different dimensions used in this section; green is used for positive values while red is used for negative. a) Original image. b) Lightness conspicuity. c) Red-green color-opponents conspicuity. d) Blue-yellow color-opponents conspicuity. e) Saliency map.

Figure 80 illustrates the calculated conspicuities for L, O_r, and O_b. For illustration purposes we show positive values in green and negative values in red; for example, dark objects near light objects have a negative conspicuity and it is shown in red. Figure 80 (e) shows the arithmetical average of the conspicuities representing the total saliency of the image. In order to compute the conspicuity map of an image, one must follow three steps: a) feature ex-

traction, b) conspicuity computation and c) normalization. A feature is the value on a given dimension in a given location, while conspicuity is the difference of the feature value of a location with its surroundings. Finally, the saliency is a combination of the conspicuity values.

5.3.1.1 Feature extraction

We use a slightly modified version of the conspicuity computation of Itti et al. (Itti, Koch, and Niebur 1998). That work computed the saliency of a location in the lightness, red-green color-opponent, blue-yellow color-opponent, and orientation dimensions. We only compute the first three dimensions by converting the image from the RGB to the CIEL*a*b* space, which already encodes the lightness and opponent colors dimensions, similar to the work of Achanta et al. (Achanta et al. 2008) (the initial RGB values are given in simplified sRGB with a Gamma of 2.2; we assume the observer at 2°, and use the D65 illuminant).

5.3.1.2 Conspicuities computation

The next step computes the conspicuities for each separate dimension. This is done by calculating the difference between a location and its neighborhood by the center-surround technique. This technique calculates the relation of a location to its surroundings by checking the difference across fine and coarse levels. Accessing finer and coarser levels of the image is done by using the built-in hardware mip-mapping, as suggested by Lee et al. (Lee, Kim, and Choi 2007), unlike the technique of Section 5.2, which used floating point textures and hence required a custom mip-map computation. The center-surround technique, as described by Itti et al. (Itti, Koch, and Niebur 1998), but modified to fit our conspicuity dimensions, is:

$$c_k = \frac{\sum_{n=2}^{n=2} \sum_{m=3}^{m=4} k_n - k_{n+m}}{p},$$

where $k \in \{L, O_r, O_b\}$ and p=6, the number of levels of coarseness being considered.

An important difference between our work and that of others is that we do not use the absolute values of the conspicuities before summing them. This allows us to keep the sign of the conspicuity; for example, if the current location (fragment) has a negative lightness conspicuity, then it is a dark location on light surroundings.

5.3.1.3 Normalization

We use a normalization that considers the global conspicuity maxima as described by Lee et al. (Lee, Kim, and Choi 2007). This has the effect of reducing non-contributing high-frequency artifacts on each dimension. The normalized conspicuity is then defined as:

$$\hat{c}_k = \frac{c_k}{\max(c_k)},$$

where $k \in \{L, O_r, O_b\}$.

The calculation of the normalization weights is a computationally demanding task. We allow the computation of these weights to be done every few frames. The number of frames is determined by the current frame rate of the system in order to maintain at least 30fps.

5.3.1.4 Saliency computation

The saliency of a location is the arithmetical average of its normalized conspicuities. The saliency at a given location is computed with the same formula given in Section 5.2.1.1.

5.3.2 Modulation of attention areas

Modulating a location means either reducing its conspicuity (in the case of context) or increase it (in the case of focus). To modulate the conspicuity of a location we must, for example, lighten or darken it, reduce or increase its “redness" or “greenness". The purpose of the technique of this section is to apply the appropriate amount of change to the image such that the information of the context is not lost. This means that the modulations are not all applied to every fragment equally. For example, some fragments might need a strong red-green modulation, but no blue-yellow modulation. Modulation is performed in three sequential steps: first, lightness is modulated, then red-green opponents, and finally blue-yellow opponents. The order of this sequence is given by the sensitivity of the human visual system to each dimension; we are more sensitive to lightness than to chromatic information, and we are more sensitive to red-green stimulus than to blue-yellow (Sangwie 1998; Spillman and Werner 1990).

5.3.2.1 Classification

Before we can modulate the image, we need a classification of the objects in the scene. This classification tells us whether we want to direct attention towards a given object (focus) or away from it (context). As detailed in Section 5.1.1, in this chapter, we assume that this classification is given through a-priori knowledge of the scene.

5.3.2.2 Modulation thresholds

The set of conspicuities encodes the difference of every location with its surroundings. However, in order to modulate conspicuities adaptively, we need a threshold for every dimension to compare a location's conspicuities to. Similar to Section 5.2.1.2, we use the average focus values as our threshold with the same notation. Throughout the rest of this section, the threshold values will be referred as t_k, where k is the given dimension.

5.3.2.3 Modulation steps

The modulation procedure is a series of analyses and adjustments. The analysis step generates information that is used by the adjustment step to verify whether further changes in a

given location are necessary. The adjustment step modifies the location in each of its dimensions separately in order to reduce its conspicuity (or increase it depending on the classification). These steps are done multiple times, from coarse to fine levels of the image pyramid by using the built-in mip-mapping capability of the graphics unit. This allows changes affecting a large region to occur early in the process, while later steps progressively refine the result. An implicit benefit of starting with coarse resolutions is that modulation of lower frequencies introduces less noticeable artifacts. Each analysis and adjustment is carried out in a fragment shader that executes in a render pass on the current image. The total number of passes necessary for modulation can be expressed as: *2 + 6 * n,* where *n* is the height of the pyramid. Two passes are necessary to convert the image to and from CIEL*a*b* space and six passes are necessary for the adjustments in the three considered dimensions and their respective analyses. The analysis step computes the conspicuity values of the input image as described in the Section 5.3.1. Figure 81 shows a flowchart with a detailed description of all the necessary steps.

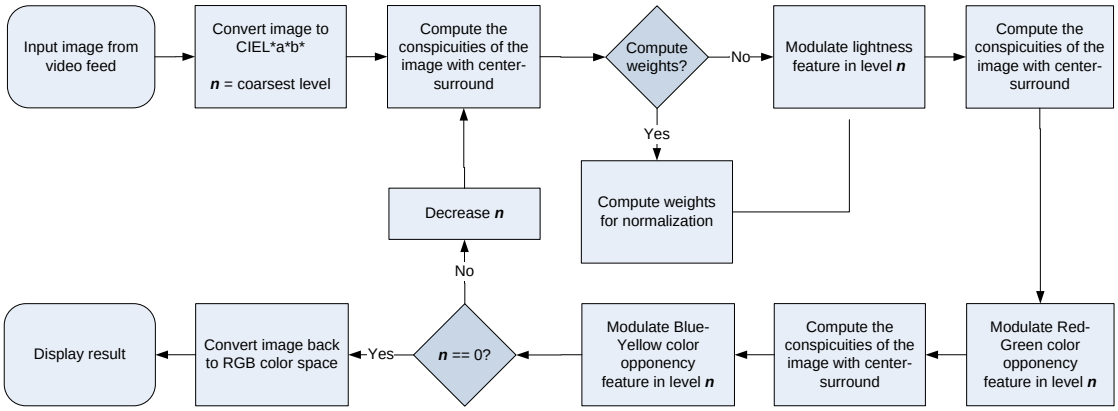


Figure 81. Flow chart of our technique. This flow chart illustrates the iterative process of our modulation technique.

5.3.2.4 Compute the modulation values to be applied

To determine the modulation value to be applied to the current location, we first verify that the location's conspicuity absolute value exceeds the given threshold. The conspicuity value is provided by the analysis step. If it does not exceed it, then no modulation is necessary and we leave the feature value unmodified. Otherwise, we compute the difference between the threshold and the current conspicuity value. This difference is then used as modulation value to be applied to the current feature. At this point, it is important to remember the roles of conspicuity and feature. A conspicuity is the difference between a location and its surroundings. We cannot modify the conspicuity directly. Instead, it is modified indirectly by changing the feature values of the location.

Let m_k be the modulation to be applied to the location, c_k the conspicuity of the location and t_k the threshold of the conspicuity, where k is the given dimension.

$$m_k = \begin{cases} 0 & c_k < t_k \\ c_k - t_k & otherwise \end{cases}$$

Before applying this modulation, a few checks must be performed in order to avoid unpleasing visual artifacts. For example, in the chromatic channels, we must also take care that the modulation should not flip the hue completely; that is, blue should never become yellow, and red should never become green, or vice versa. This is done by preventing a flip on the sign of the feature value, a positive value (e.g., red) cannot become negative (e.g., green).

5.3.2.5 Coherence

The modulation process seeks to reduce the amount of changes in the original image. This naïve definition, however, only considers the appropriate values for each location without regard to the global coherence of the image. This then may result in noise artifacts, typically chromatic, on the final image as illustrated in Figure 4. Such artifacts occur when two spatially close locations are matched to different modulation values by our technique. This only happens when the conspicuity of a location is increased (focus) and the original chromatic values are close to zero. To suppress these artifacts, we compute the average between the modulation computed at the immediately coarser pyramid level and the current level. A side effect of this technique is that the strength of the modulation is reduced.

Figure 82. Spatial coherence. This figure illustrates the problem arising from the emphasis of contrast in the focus area. Tungsten light on the metallic surface of this car model caused these particular artifacts. Notice the red or green dust in the middle image. a) Original image. b) Image affected by naive conspicuity enhancement. c) Image after applying our spatial coherence technique.

As stated before, the computation of the normalization weights is amortized over several frames, depending on the current frame rate. If the amortization period is too long, the changes on normalization weights may be drastic. This can introduce "jumping" modulation artifacts between two adjacent frames. We therefore compute the weight and thresholds

using a sliding average. Once our modulation value has been computed and all checks necessary to ensure that no drastic changes occur (either spatially or temporally), we can apply this value to the location. To decrease the conspicuity of the context, we merely subtract the change value we computed from the given location's feature.

This has the effect of reducing the distance between the current conspicuity value and the threshold. To increase the conspicuity of a location, instead of subtracting, we simply add the change value we computed to the given location's feature. This has the effect of increasing the distance of the current conspicuity and the threshold.

Let f_k be the feature value of the location and f_k' the modulated feature value and m_k the modulation to be applied where *k* is the given dimension. Then,

$$f_k' = \begin{cases} f_k - m_k & if\ the\ location\ is\ marked\ as\ context \\ f_k + m_k & if\ the\ location\ is\ marked\ as\ focus \end{cases}$$

5.3.3 Validation

To test the performance of our second saliency modulation technique, we carried out two formal user studies. The goal of the first study, performed with an eye tracker, is to test whether our technique can direct the visual attention of the user towards the regions of the videos that we designated as focus. We will refer to the first study as the *attention study*. The goal of the second study is to identify the amount of modulation that we can perform on the videos without the users noticing. We will refer to the second study as the *awareness study*.

Our hypotheses for the attention study are:

- **H4.** The duration before the first fixation on the focus regions will be smaller for the videos modulated with our procedure than for the original unmodified videos.
- **H5.** The fixation time (i.e., sum of durations of all fixations on the focus) in the focus regions will be higher for the videos modulated with our procedure than for the original unmodified videos.
- **H6.** The percentage of participants that have at least one fixation on the focus region will be higher for the images modulated with our procedure than for the original unmodified videos.

The awareness study does not have any hypotheses because the goal is to learn information from the population. Thus, the goal is to obtain the highest modulation threshold with which we can modulate the image without the users being aware of it.

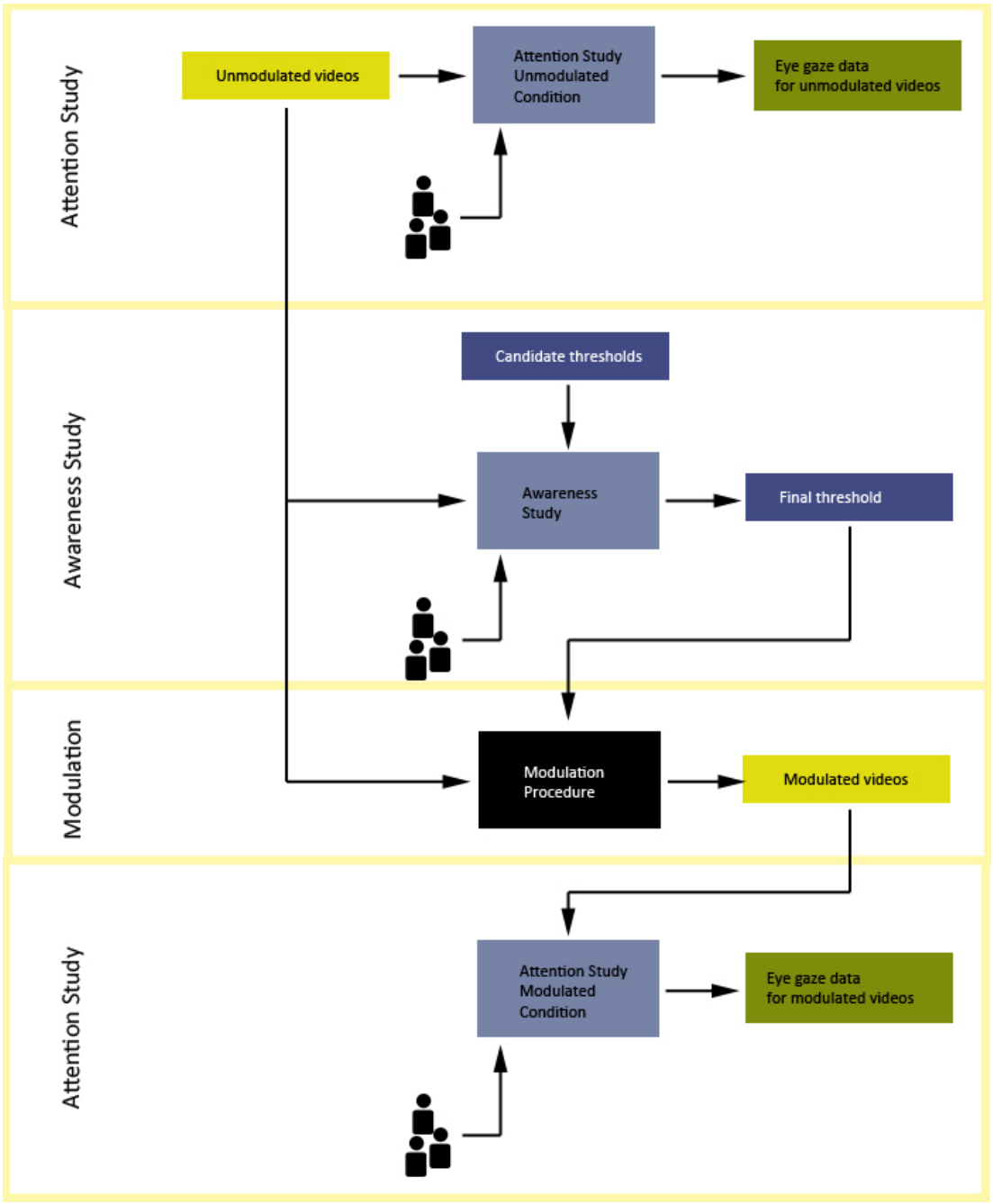


Figure 83. Setup for studies. This figure illustrates the multiple steps and the flow of information for the awareness and attention studies, along with the modulation procedure step.

5.3.3.1 Experiments description

The two experiments were orchestrated together but with separate populations. The attention study had two conditions and was performed between subjects. The two conditions were performed separately one month from each other. The awareness study was performed after the first condition of the attention study. The final set up of the studies has a rather complex data flow. To summarize this, Figure 83 illustrates the multiple steps and the

order in which they are taken. The following sections will explain in detail each of the steps and stimulus information in our setup.

Stimulus videos

We recorded twenty videos with a digital video recorded for this user study. These videos were filmed indoors, outdoors, during daytime, during nighttime, with moving objects (cars), without moving objects, while walking, while standing still and panning, etc. The goal was to have a manageable variety of videos that represented day to day situations. The only restriction was that there should not be any human body parts (hands, legs, and specially faces) in the videos due to the high attention sink they represent. However, pictures of people such as those from billboards were allowed.

The videos were recorded with a Samsung Xacti at 1280×720 with 29.97fps. They were then used unprocessed and uncompressed for the unmodulated versions of the studies. There was no cropping, digital steadying, contrast enhancement or any other type of manipulation. Each video had a duration of roughly 10 seconds.

Attention study

The goal of the attention study is to verify with an eye tracker whether we can successfully direct the visual attention of the participants with our saliency modulation technique. Similar to the study from Section 5.2.3, we assume that the human visual attention can be characterized by the eye gaze. The study was divided in two conditions, unmodulated and modulated.

The goal of the unmodulated condition was to obtain the regions in the unmodulated stimulus videos that were visually unattended by the participants. We define the unattended regions as those that had less than five fixations by less than twenty percent of the population.

The unmodulated condition was performed on twenty participants (14 female, 6 male) between 24 and 52 years old ($\bar{x}$=31.4). Half of the participants had normal vision while the other half had corrected to normal vision. All participants had no known color sensitivity deficiencies; which was confirmed by an on-screen Ishihara color test. Participants were compensated with a gift certificate from a popular online shop.

Once the participant arrived in the test she was welcomed and allowed to read an informed consent form that had to be signed before proceeding. After this, she was provided with the following instructions:

You will sit in front of a computer screen. We will display a series of short video clips. All you have to do is look at the clips. That's it!

This test is divided into two parts so you can have a break in between.

Your eye gaze will be tracked with a non-wearable system. It will be using an infrared camera and light placed in front of you. Infrared light is invisible to the eye and poses no harm to you.

Care was made not to mention the number of video clips in order to avoid counting (which would trigger a top-down task). After the instructions were read, we asked the participants to verbally repeat the instructions. It was then verbally emphasized by us that she did not have any task and that all that was required was to see the clips. The eye tracker was calibrated for each participant before the stimulus was presented. Each participant saw once each of the twenty unmodulated videos. The presentation of the videos was randomized. Between videos, a blank slide was shown for 2000ms.

The goal of the modulated condition was to obtain the regions in the modulated stimulus videos that were visually attended by the participants. The modulation threshold was obtained by the awareness study. Please refer to Section 5.3.2.2 for more details on the modulation thresholds. The eye gaze data was then analyzed to verify that we satisfied the hypotheses given earlier (see Section 5.3.3.2).

The modulated condition was performed on twenty participants (6 female, 14 male) between 25 and 42 years old ($\bar{x}$=32.1). These participants were different from those of the unmodulated condition. Fifteen participants had normal vision while five had corrected to normal vision. All participants had no known color sensitivity deficiencies; which was confirmed by an on-screen Ishihara color test. Participants were compensated with a gift certificate from a popular online shop.

The participants went through the same procedure as those participants from the unmodulated condition. They also received the same instructions. Once again the eye tracker was calibrated before the videos were shown. Each participant saw once each of the twenty modulated videos. The presentation of the videos was randomized.

Awareness study

The overall goal of these studies was not only to successfully direct the visual attention of the participants. That was already demonstrated in Section 5.2.3. Our goal was to direct the attention of the participants without letting them notice that the videos have been manipulated. This poses a difficult problem, in order to verify that the participants do not notice a difference between the modulated and unmodulated versions, they must be actively checking for visual manipulations in the videos. This is a goal based task. Such a task is known to modify the gaze path of participants and suppress stimulus based attention. We cannot record stimulus based eye gaze data if we give the participants a task. Thus, the participants

of the modulated condition of the attention study cannot check for manipulations in the video at the same time.

This is why the awareness study was conceived. The goal of the awareness study is to find out the highest modulation threshold with which we can modulate the videos without having the participants notice. This modulation threshold was used to modulate the videos used for the modulated condition of the attention study.

The modulation threshold is actually a collection of three values, one for each: lightness red-green opponents and blue-yellow opponents. We express the thresholds as floating point values in the [0...1] range. We discretized this range into a limited set of evenly spaced samples, fitting a natural logarithmic curve. Additionally we took the extreme values zero and one, for a total of nine modulation thresholds.

The threshold values for each the three dimensions, lightness, red-green and blue-yellow were the same. The reasoning behind this is the same that led to the combination of the conspicuities to generate the saliency as suggested by Itti et al. (Itti, Koch, and Niebur 1998).

We performed two pilot studies to investigate the appropriate modulation threshold we should use. However, the results were not entirely agreeable, so we performed a larger more formal study. We now present these studies in sequence.

Likert scale based pilot

The first pilot was conducted on three participants (all male with ages 28, 33 and 35). All had normal or corrected to normal vision and had no color sensitivity deficiencies (confirmed by an Ishihara test). This pilot was based on the idea that participants should judge the stimulus and provide a score in a Likert scale.

The stimulus was a series of videos, each modulated with one of the thresholds (notice that threshold zero actually means no modulation at all). Each video was randomly modulated with only one of the thresholds. There were in total eighteen videos to be shown to the participant. They were shown in two sets of nine videos, each in order to let the participant have a short break. The following instructions were given to the participants:

You will sit in front of a computer screen. We will display a series of short video clips for you (about 10 seconds each). We ask you simply to look at the videos.

Your task is to judge how natural the videos look. After each clip we will ask you to verbally give a score to the video.

The score is as follows. In the following scale, 1 (natural) means that the video is as it came from the camera, and 7 (unnatural) means it has been manipulated to an extent such that it feels unnatural.

Please give a score to each video.

1	*2*	*3*	*4*	*5*	*6*	*7*
natural			*somewhat unnatural*			*unnatural*

After the instructions were read, we asked the participants to verbally repeat them. They were then verbally reminded the values of the score in the Likert scale. The eye tracker was not used for this pilot. Each participant saw once each of the eighteen stimulus videos. The presentation of the videos was randomized. Between videos, a slide with the message "score?" was shown at which time the participant had to provide the score. The participants had to see the clips for the whole duration and could not provide a score while the clip was playing.

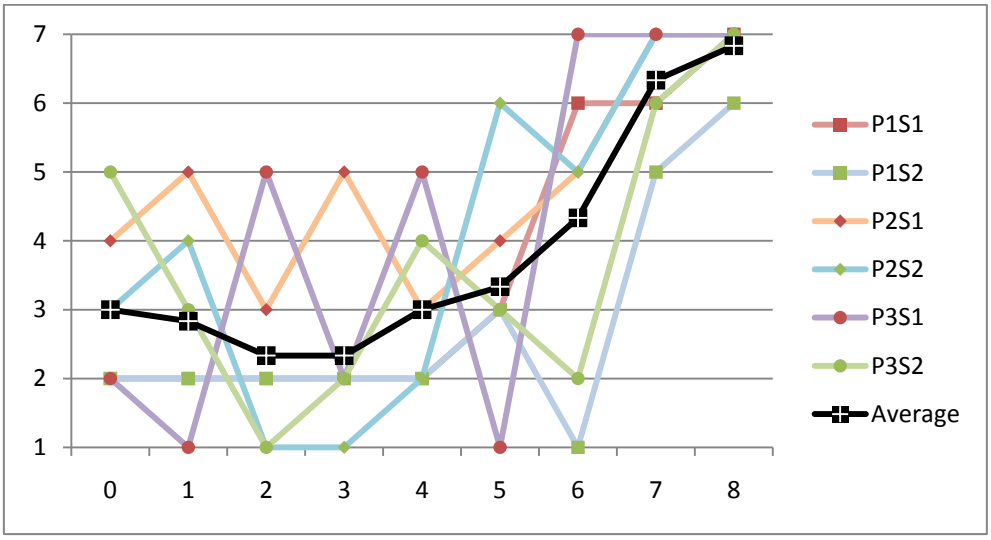


Figure 84. Responses of the first pilot. These are the actual responses given by the participants in the first pilot study. Each participant saw 18 videos, two modulated for each of the nine threshold levels. Notation "P1-S2" means "Participant one, set two".

Figure 84 is a line graph shows the actual responses of the participants. Additionally, the average response across all participants is also plotted (in black). Horizontal axis denotes the modulation threshold while vertical axis denotes the given response. Notation "P2S1" denotes "Participant 2, set 1". Notice that every participant has two line graphs, each corresponding to one stimulus set of nine videos.

We did not perform any type of statistical analysis in this set due to its small data size. One can see, however, that the higher the modulation threshold the higher the score given by the participants. Thresholds from zero to five scored below “somewhat unnatural”. In fact, thresholds zero and four both had on average a score of three.

Change blindness based pilot

The second pilot was conducted on three participants (two males, 1 female with ages 28, 29 and 24). All had normal or corrected to normal vision and had no color sensitivity deficiencies (confirmed by an Ishihara test).

The goal of this pilot was to verify whether the participants notice a difference between modulated and unmodulated images. The images were randomly selected screenshots from the stimulus videos. These were presented in pairs while at the same time inducing change blindness following the setup suggested by Rensink et al. (Rensink, O'Regan, and Clark 2000).

For each pair the images were presented in the following order:

- First image shown for 240ms
- Blank image shown for 320ms
- First image shown for 240ms
- Blank image shown for 320ms
- Second image shown for 240ms
- Blank image shown for 320ms
- Second image shown for 240ms
- Blank image shown for 320ms
- Repeat

We had nine change blindness modulation setups, one for each threshold being considered. We modulated two images for each of these nine modulation thresholds. Similar to the previous pilot, the total data set was eighteen image pairs. Remember that modulation threshold zero means no modulation. Thus the pair “zero-three” means that the image without modulation was shown for 240ms followed by a 320ms blank image, then again the zero modulation image, then a blank image, then the image modulated with threshold three, then blank, then the modulated image again and so on.

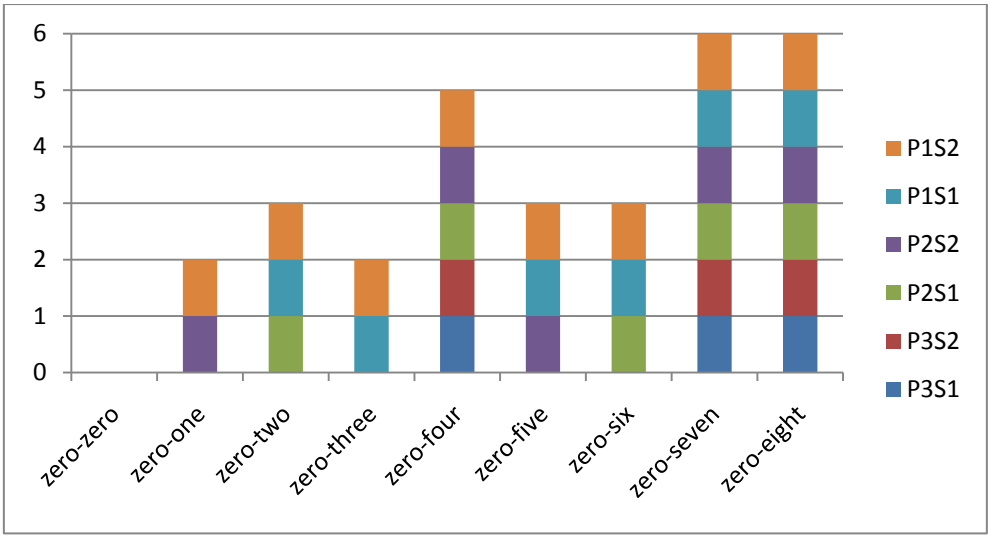


Figure 85. Responses of the second pilot. These are the actual responses given by the participants in the second pilot study. Each participant saw 18 image pairs, two modulated for each of the nine threshold levels. Notation "P1-S2" means "Participant one, set two". Notation "zero-five" means that the pair used was the same image in an unmodulated and modulated by threshold five.

The following instructions were given to the participants:

You will sit in front of a computer. A few sets of two alternating images will be shown on the screen. Your task is to decide whether you can see a difference between the two images.

You will have up to one minute to make your decision while the images automatically alternate. At the end of this minute you will be asked for an answer. You may also give an answer before the minute is up if you desire.

Please state your answer in the following form:

- *Yes. I can see a difference between the two images*
- *No. I cannot see a difference between the two images*

A total of eighteen pairs of images will be shown to you; these will be presented in two sets of nine pairs each with a five minute break in between.

After the instructions were read, we asked the participants to verbally repeat the instructions. The eye tracker was not used for this pilot. Each participant saw once each of the eighteen image pairs. The presentation of the image pairs was randomized. Between image pairs, a slide with the message "score?" was shown at which time the participant had to provide the score. As suggested by Rensink et al., the total duration of the change blindness stimulus was presented for 60 seconds for each pair (Rensink, O'Regan, and Clark 2000). The participants, however, had the possibility to give a score before the end of the stimulus.

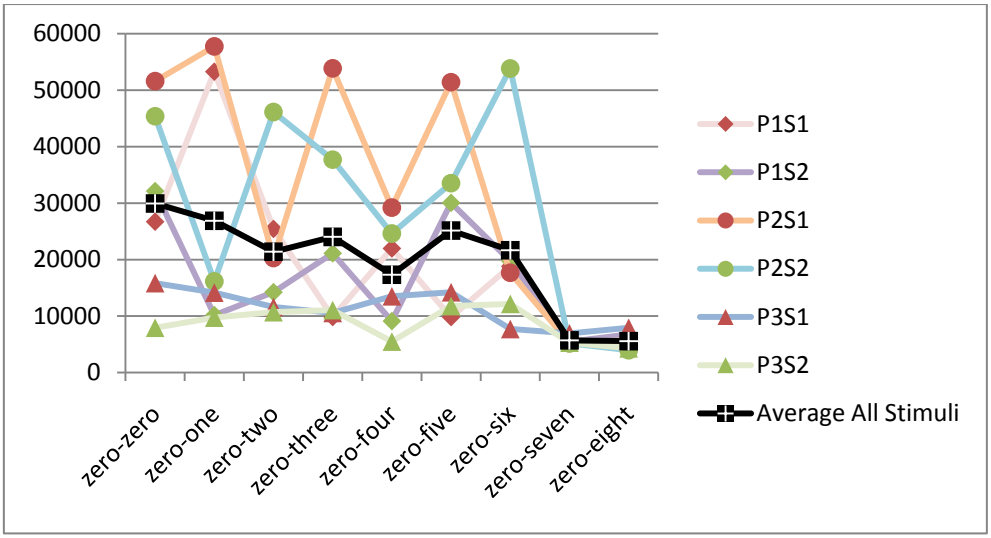


Figure 86. Response times of the second pilot. These are the number of milliseconds that passed before the participant emitted a judgment on the image pairs. Notice how it became increasingly easier to judge the image pairs.

Figure 85 shows the responses of this study in a stack bar form. We interpret each affirmative response as a value of 1. We interpret each negative response as a value of 0. As it can be seen the pair "zero-zero" was always correctly judged as being without modulation by all participants (it never received an affirmative response). Pairs "zero-seven" and "zero-eight" were also always correctly judged as being modulated (6 affirmative responses). Intriguingly, pair "zero-four" was graded higher than pairs "zero-five" and "zero-six".

Figure 86 shows the line graph plotting the response times of the second pilot. Each point in the graph tells how many milliseconds passed since the stimulus started until the participant gave a judgment to the image. Notice that stronger modulated images were easier to judge.

Formal awareness attention study

In summary, out of the nine original threshold candidates the first pilot suggested the use of values four or five. The second pilot suggested the use of value three. These results are not entirely agreeable. Thus, we decided to perform a more formal study based on the first pilot in order to find out what threshold value we can use. We did this study based on the first pilot because we believe its setup matches better the conditions of the attention study. Remember that in the attention study the participants only see either the unmodulated or the modulated version of the videos, but not both.

This study was performed on sixteen participants (12 male, 4 female) between 18 and 35 years old ($\bar{x}$=27.8). The participants of this study were different from all the previous participants. Four participants had normal vision, while twelve had corrected to normal vision. All

participants had no known color sensitivity deficiencies; which was confirmed by an Ishihara color test. Participants were compensated with a gift certificate from a popular online shop.

For this study we decided to use the actual stimulus videos from the attention study. Thus, the three candidate thresholds (three, four and five) and the control (no modulation) times the twenty stimulus videos gave a total of eighty video-threshold pairs. We arranged the videos in such a way that each video-threshold pair was seen by four participants. Each participant saw each video with a semi-randomized modulation threshold. No participant saw the same video twice with different modulation thresholds.

Once the participant arrived in the test she was welcomed and allowed to read an informed consent form that had to be signed before proceeding. After this, she was provided with the same instructions as the first pilot. After the instructions were read, we asked the participant to verbally repeat the instructions. She was then verbally reminded the values of the score in the Likert scale. The eye tracker was not used for this pilot. Each participant saw once each of the twenty video-threshold pairs. The presentation of the videos was randomized. Between videos, a slide with the message "score?" was shown at which time the participant had to provide the score before proceeding. The participants had to see the clips for the whole duration and could not provide a score while the clip was playing.

We then proceeded to analyze the resulting data. We considered the four modulation thresholds (zero, three, four and five) as related samples. We then conducted three Wilcoxon signed tests for two related samples. The idea is to compare to the ground truth whether users noticed a significant damaging of the videos. Thus, our resulting pair samples were zero-three, zero-four and zero-five. The raw responses from the participants are provided as a part of the Appendix (Section 8.1). We applied a Bonferroni correction to account for the number of pair samples and keep our alpha levels below 5%. The analysis showed that there was no statistically significant difference between any of the pairs. This indicates that to the general population it was impossible to distinguish which videos had been modulated and which had not.

These results seem to be empirically in line with the results of the first pilot. For us, this means that we can use any of the three thresholds safely without the users noticing that there is any modulation taking place. We decided to take a conservative approach and use threshold four for our modulation procedure.

Figure 87. Comparison between modulated and unmodulated video. (Top) Shows a frame from one of the unmodulated input videos. (Bottom) Shows a frame from the same video after modulation. When compared side by side, one may work out specific differences in the frames, but looking at the modulated version in isolation makes it hard to spot any changes.

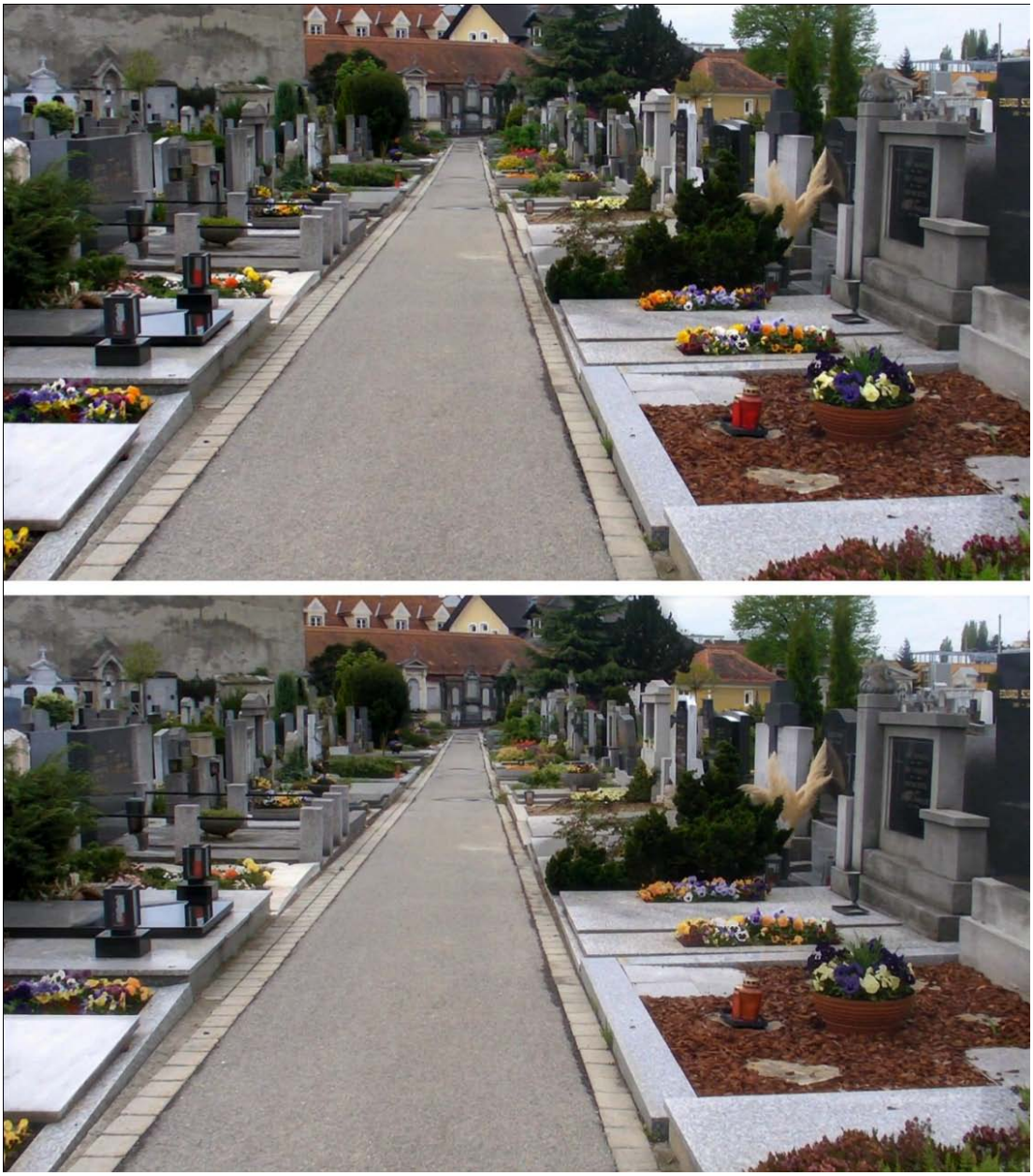


Figure 88. Another comparison between modulated and unmodulated video. (Top) Shows a frame from one of the unmodulated input videos. (Bottom) Shows a frame from the same video after modulation.

Modulation

We obtained the unattended areas on the twenty unmodulated videos with the unmodulated condition of the attention study. We then obtained the threshold level with which we can modulate the videos without participants of the awareness study noticing. With this information, we modulated the stimulus videos in order to enhance the visual salience of the unattended areas. This was done with our saliency modulation technique from Section 5.3.2. This resulted in a set of twenty modulated stimulus videos. These stimulus videos were the ones used for the modulated condition of the attention study. The flow of this information can be seen in Figure 83.

Figure 89. Detailed comparison. This figure illustrates in detail the changes applied to the input video. Notice that the modulated version of the focus is slightly more saturated while the context has faintly less contrast. In particular the front light of the scooter is not as conspicuous as in the unmodulated version. (Right) An illustration of the position of the mask.

Figure 87 and Figure 88 illustrate related images from frames of the input videos. The images on top were obtained from the unmodulated video (first condition). The images on the bottom were obtained from the video after our modulation procedure (second condition). Notice how the changes are barely perceptible if one is allowed to compare both conditions side by side. If, however, one is only allowed to see the modulated video, the changes become almost imperceptible. Figure 89 shows a more detailed comparison of the changes of Figure 87. Observe that the focus after modulation has slightly more vivid colors and more contrast, while the context is less contrasting and the colors are slightly duller. The image on the right illustrates the mask used to denote the portion to which we want to direct the attention.

Apparatus

The tracking device was an SMI desktop-mounted eye tracker, operating at 60 Hz. The stimuli were presented on a 19" monitor at a 70cm distance from the participant. The resolution of the videos was 1280×720, and all were presented without resizing in order to avoid interpolation by the graphics unit. All studies were performed in an empty office with the lights off, closed windows and closed doors in order to minimize any attention distracters.

5.3.3.2 Analysis of the attention study

Analysis is performed with independent samples t-tests whenever our data satisfied the condition of normality and with Mann-Whitney U-tests otherwise. A Shapiro-Wilk test indicated that the data for our hypotheses H4 and H6 satisfy normality. However, the data of H5 does not. We adjusted the α levels (Bonferroni) to ensure a level of 5%. Both the t-test and the Mann-Whitney U tests are one tailed.

The first hypothesis of this study we formulated was: (H4) The duration before the first fixation on the focus regions will be smaller for the videos modulated with our procedure than for the original unmodified videos. This means that we would be able to attract the visual attention of the participants faster with our modulation technique.

The results of the one-tailed t-tests indicate that the mean values of the second condition (modified) are significantly smaller than the mean values of the first condition (unmodified), $t(35) = 2.916$, $p < .01$. That is, the mean duration before the first fixation on the focus regions for participants in the second condition ($M = 9231.58$, $SD = 468.16$) was significantly smaller than that of the participants in the first condition ($M = 9638.86$, $SD = 363.10$).

The second hypothesis of this study we formulated was: (H5) The total fixation time (i.e., sum of durations of all fixations on the focus) in the focus regions will be higher for the videos modulated with our procedure than for the original unmodified videos. This means that we would be able to retain the fixations of the participants for a longer time with our modulation technique.

The results of the one-tailed Mann-Whitney U-test indicate that there is no significant difference in the total fixation time between the unmodified and the modified conditions, $p = .03$. Despite the lack of normality of the data for this hypothesis and due to its robustness, we also performed a t-test on this data. The results of the one tailed t-tests confirm that there is no significant difference in the total fixation time between the unmodified and the modified conditions, $t(35) = -2.117$, $p = .02$. That is, the mean total fixation time for participants in the second condition ($M = 43.31$, $SD = 24.32$) was not significantly different from that of the participants in the first condition ($M = 26.79$, $SD = 22.82$).

The third hypothesis of this study we formulated was: (H6) The number of participants that have at least one fixation on the focus region will be higher for the images modulated with our procedure than for the original unmodified videos. This means that a larger portion of the population would be visually drawn to the focus regions with our modulation technique.

The results of the one-tailed t-tests indicate that there is no significant difference in the number of participants that had at least one fixation between the unmodified and the modified conditions, $t(35) = -2.028$, $p = .05$. That is, the mean number of participants with at least one fixation in the second condition ($M = .13$, $SD = .08$) was not significantly higher from that of the participants in the first condition ($M = .08$, $SD = .06$).

As can be seen, hypothesis H4 is statistically significant; however, we are unable to find statistical differences on hypotheses H5 and H6. We further examined the gaze data of our participants to try to find consistent failures in our modulation procedure. By visually analyzing the heat maps of our videos on the second condition, we found what seemed to be a consistent pattern where our modulation procedure failed. This is, whenever the camera panned directly away from the focus of interest the technique seemed to be unable to attract fixations. This does not happen on videos where the camera is static, or whenever the panning is not directly away from the focus regions. Subsequently, we filtered out the information from focus regions that fit this criterion (5 regions out of 29 are excluded). Then we proceeded again to perform the analysis.

Once again, we performed a Shapiro-Wilk test to verify the normality of our filtered data. The results indicated that the filtered data for our hypotheses H4 and H6 satisfy normality but the filtered data of H5 does not.

On the filtered data of hypothesis H4, the results of the one-tailed t-tests indicate that the mean values of the second condition (modified) are significantly smaller than the mean values of the first condition (unmodified), $t(35) = 3.386$, $p < .01$. That is, the mean duration before the first fixation on the focus regions for participants in the second condition ($M = 9126.98$, $SD = 499.44$) was significantly smaller than that of the participants in the first condition ($M = 9616.66$, $SD = 352.59$).

On the filtered data of hypothesis H5, the results of the one-tailed Mann-Whitney U-test are in the expected direction and statistically significant, $z=-2.576$, $p< .01$. The second condition had a mean rank of 23.23 while the first condition had a mean rank of 14.03. Despite the lack of normality of the data for this hypothesis and due to its robustness, we also performed a t-test on this data. The results of the one-tailed t-tests confirm that there is a significant difference in the total fixation time between the unmodified and the modified conditions, $t(35) = -2.659$, $p < .01$. That is, the mean total fixation time for participants in the second condition

(M = 49.52, SD = 26.04) was significantly higher than that of the participants in the first condition (M = 27.65, SD = 23.55).

On the filtered data of hypothesis H6, the results of the one tailed t-tests indicate that the mean values of the second condition (modified) are significantly higher than the mean values of the first condition (unmodified), $t(35) = -2.478$, $p < .01$. That is, the mean number of participants with at least one fixation in the second condition (M = .15, SD = .09) was significantly higher than that of the participants in the first condition (M = .08, SD = .06).

As can be seen, all three hypotheses are found to be statistically significant on the filtered data. This is, if the camera is not panning away from the focus region, we can draw the eye gaze of participants significantly sooner and retain it significantly longer with our modulation technique. Additionally, a significantly higher number of participants had at least one fixation in the focus regions.

5.3.4 Discussion

The goal of these user studies is twofold. First, we obtained the maximum modulation threshold that can be applied to the input videos such that participants are not able to tell whether the video is manipulated or not. Then we took that threshold and modulated the videos in order to direct the attention to areas previously unattended by the participants.

As can be seen from the analysis, our modulation technique can effectively draw the attention of the participants to the focus region in the modulated condition. The average duration before the first fixation on the focus regions was significantly smaller. In those videos where the camera does not look directly away from the focus region we can also retain the visual attention of the participants for a significantly longer time. In the same videos, the number of participants with at least one fixation on the focus regions of the modulated videos was significantly larger than those of the unmodulated videos.

This implies an important finding: That the biggest sink of attention seems to be the camera motion. Although we expected that the motion of objects in the video would a high attractor, this was not the case. Instead the way in which the filming camera moved is a bigger attractor and our modulation technique is incapable of suppressing it for hypotheses H5 and H6. One must also remember that these videos do not include any body parts, such as faces, since these are known to be extreme attention sinks.

It is difficult to illustrate the accumulated fixations of a region on a video feed. The fixations on a region are spread out throughout the entire duration of the video clip. Nevertheless, Figure 90 illustrates one frame of one video in both conditions in which the effects of the modulation technique are clear. The image on the top comes from the unmodulated video

while the image on the bottom comes from the modulated video. A white outline denotes the position of the focus region (middle left). This example is chosen specifically to illustrate this example. In the general case, however, there is no guarantee that the effect will be this apparent throughout the entire duration of the video.

The technique is not always effective for each of the video clips, or for each of the participants in the tests. But in the general case, one can confidently say that the technique will draw a first fixation faster than without modulation. In the cases where the camera is not moving away from the focus regions the number of participants with at least one fixation in the focus region will be significantly higher, and the fixation time will also be significantly higher. Thus, we can state that the attention direction technique by saliency modulation introduced in Section 5.3 was successful.

Figure 90. Heatmaps of the user studies. (Top) Heatmap resulting from the unmodulated condition. (Bottom) Heatmap resulting from the modulated condition. The focus of attention is marked with a white rectangle. This is a handpicked example chosen to illustrate the effect of our modulation technique. In the general case, however, the fixations would be spread out throughout the entire video duration and would not be condensed on a single frame.

Chapter 6

Conclusion

AR allows us to interact with computing systems in a more natural way than traditional interfaces. AR is not a piece of software or hardware; rather, it is an interface paradigm that enables us to blend physical and virtual information and experience it in an intuitive way. Like many other computer science disciplines, AR suffers from an unbalanced use of resources. While the central portion of data (the focus) of any task is thoroughly exploited, any information that provides a situational perspective (the context) is simply left untouched and disregarded.

6.1 Reflection on the proposed techniques

This dissertation tries to make the point that AR, just like any other computer science discipline, should take advantage of contextual information. Throughout the previous chapters, we have given examples and encouragement on different ways in which this can be achieved. Arguably, the given examples are quite varied and propose new and broader ways in which to look at data. Throughout the developing of this work, the author has found it fruitful to look away from the blinding importance given to the focus information and instead looking for ways in which the context can help solve the problem.

But, although it is likely that contextual information can aid better perceptual and visual AR experiences, this may not always be necessary or even possible. One of the constant side effects of the techniques presented in this dissertation is that most of them cause some computational overhead, in some cases of considerable weight. Although all of the examples can be improved with better implementations, in the end the extra computational effort can never be eliminated. Thus, the responsibility to whether the applications should be made context-aware lies on the application designer. This decision will always be application and task dependent, and one must be aware that it might not always pay off.

Chapter 3 presented a series of techniques that exploit the existence of contextual information in the scenegraph data structure of the AR application. The presented examples generate styles that affect the final visual result of specifically tagged contextual subgraphs. This is done either by modifying visual parameters such as color or scale, as well as by applying magic lenses techniques or even procedurally modifying the generation of geometry. The final techniques provide a wide range of styling possibilities at the cost of traversal overhead. For any of the techniques of that chapter, a checking procedure and an online strap-

ping of predefined nodes had to take place during rendering. This means that any context-aware node has to be traversed twice for simple styling and three times for magic lens styling. For context-sensitive scenegraph traversal, we used Reitmayr and Schmaltieg's implementation of a custom scenegraph state element that allows the propagation of textual tags during traversal (Reitmayr and Schmalstieg 2005). In order to provide the scenegraph with these textual attributes, a transcoding pipeline from the source GIS database is necessary. This pipeline converts data from geospatial databases onto our required scripting format. Such geospatial data is often two-dimensional; thus, the pipeline must transcode raw two-dimensional geospatial data into three-dimensional models suitable for our rendering engine. Transcoding is not simply a one-to-one conversion from one format to another, and it does not require only geometrical data. Transcoding the geospatial database information's semantic attributes into visual primitives entails information loss. The amount of retained information will affect the speed of our rendering, and therefore we must find the right amount of information to preserve. If we discard too much information, we cannot use it to interact with the user later in the pipeline. If we discard too little, we have to reinterpret the semantics at runtime, which increases overhead and adversely affects performance.

Chapter 4 introduced a collection of techniques based on the usage of contextual information in order to enhance the perception of depth in AR scenes. These techniques consider and exploit spatially occluding information in order to provide more careful augmentations. Traditional AR occlusion management techniques do not truly consider the occluding information and instead rely on extra augmentations (such as sign posts) to ameliorate the problem. In contrast, our techniques carefully analyze occluding information and retain those portions that are considered of high value, this is referred to as "the focus of the context". This brings up the question of how this analysis is done, what its performance impact is and how effectively it works. Four techniques were presented, an analysis of an artificial model, an analysis of the video feed, the usage of a predefined mask and the usage of a procedurally generated mask. The predefined mask is the least computationally expensive technique, but it requires a detailed knowledge of the scene before deployment. The other three techniques are computationally expensive, requiring edge detection, haloing and noise computations on the fly. In order to reduce the performance impact, all of the techniques are implemented on the GPU, but this effectively reduced the number of systems in which they can run.

Chapter 5 proposed the concept of using spatially surrounding information in order to direct the visual attention to a focus region. Two techniques were presented that differ on the amount of control they allow and the color space in which they work. Both techniques were formally tested with extensive user studies, confirming that they indeed are capable of shifting the eye gaze of the participants. Moreover, the technique from Section 5.3 does so in a

way that is imperceptible to the participants; that is, by manipulating contextual information we can direct the human visual attention without participants consciously noticing.

The techniques, however, are computationally demanding, requiring up to 24 rendering passes on the GPU. Although this can be reduced, the fact remains that the techniques require an intensive analysis and modulation. The normal trend of continuous computational power increase suggests that soon these techniques could be implemented on mobile devices.

Just like most AR applications, these techniques are dependent upon tracking information. In a live scenario, the effectiveness of the attention direction techniques will be heavily influenced by the quality of the tracking and the amount of detail in the virtual environment. A poor model of the shape of the object to which we want to direct the attention plus a low confidence on the tracking pose will result in an ineffective attention direction.

6.2 Future work and closing remarks

As mentioned before, this dissertation touches on multiple aspects of an AR application. The common goal is to enhance the visual results and experience of the users. One proposed way to achieve this is by styling the data structure of the application. Scenegraph styling is hardly a new topic; it has been performed in different forms and with different target data structures for many years. For example, in 1983, Beach and Stone used the concept of style sheets for controlling a uniform look over a set of illustrations called *graphical style sheets* (Beach and Stone 1983). Nevertheless, the usage of contextual attributes to drive the styling of information is a more recent development. Chapter 3 presented several techniques that exploited tagged contextual information in order to influence the visual result of the AR application. The pervasive assumption is that the scenegraph contains enough textual attributes and that mappings between the attributes and the styles exist. The effort concentrates on the transcoding step that converts information from unknown sources into a format that can be understood by the rendering engine. Moreover, the question remains on how much information should be preserved and in what detail. Further research must be carried out on two aspects: an effective transcoding pipeline, and more efficient traversing technique for the context-sensitive scenegraph.

Scenegraph management is only one topic discussed in this dissertation; occlusion management has been a central topic of research since the beginning of AR. This will continue to be subject of investigation, and new and better ways will be developed. The latest AR conferences and journals already show new ideas and proposed solutions on how to address this problem. In earlier years, the researched solutions were often based on optical see-through systems. These, however, have not become as widely accepted as video see-through systems.

In the future, a further shift to small screen devices (such as mobile phones) will be necessary. The limited screen real estate and poor camera quality is a challenge that sooner or later will have to be faced. The work presented in Chapter 4 and in particular that of Section 4.5.1 provides already a possible direction to address this. Implementation in this case is not an actual issue, but thorough validation with user testing is imperative. Although the presented techniques empirically seem to be solid, no scientific claim can be made on their effectiveness, unless formal and strict testing is performed. The path ahead, thus, is twofold: First, a shift in the target devices towards mobiles platforms and second, rigorous validations of the presented techniques.

The most clear immediate future work is the optimization of the attention direction technique from Chapter 5. The manipulation of salience values can be seen as an energy minimization problem for which there are many possible solutions. In particular, the GPU based total variations work presented by Pock et al. (Pock, Grabner, and Bischof 2007) presents an interesting alternative to our multi-pass rendering technique. The goal of the saliency modulation technique is to direct the human visual attention to specific portions of the scene in a way that is imperceptible to the users. This can be stated as an energy minimization function, where the constraint is that the salience difference has to exceed a certain threshold (see Section 5.2.1.2), but the changes to the original image are minimal. This would additionally reduce the light hazing problem of the current techniques. Moreover, we would eliminate the small amount of hue shift present in the current techniques. Paired with high quality tracking, this would prove to be a powerful attention direction technique.

Although we have thoroughly tested the attention direction with a total of two studies involving 86 participants, further testing of the technique during live situations, instead of prerecorded videos and pictures is necessary. All of our studies have concentrated on laboratory-controlled environments, where we have complete control of the setup, including the sitting position of the participant, her distance to the screen, the amount of environmental lighting and so on. True real-life scenario testing is thus recommended to more convincingly present evidence of the effectiveness of the technique. This would have to be carried out with mobile eye tracking devices and a powerful mobile computer capable of doing the modulation on the fly.

This dissertation was set out to encourage the usage of contextual information in AR environments. It did so by presenting a series of techniques that addressed varied problems, ranging from scenegraph styling to human visual attention. All of the presented techniques are successful in attaining their goal. The author of this dissertation hopes that, after reading this dissertation, the reader will look at the problems surrounding AR applications in a different way, trying to find new solutions provided by the usage of context.

Chapter 7

Acknowledgements

I would like to thank everybody without whom this dissertation would not have been possible. First and foremost, my thanks go to my supervisor, Dieter Schmalstieg, for providing me with the guidance and direction necessary to complete this work. I thank him for the encouragement and challenges he made me see throughout all these years. I would also like to thank Steven K. Feiner for taking the mantel of advisor for my dissertation, and for showing me different and more rigorous ways to look at my own work.

I would like to thank all of my colleagues at the Graz University of Technology for all the support and camaraderie these years. Special thanks go to Denis Kalkofen, Gerhard Schall and Eduardo Veas for being my team mates and co-developers of many AR applications. Likewise, I would like to express my gratitude to Ernst Kruijff who gave me the opportunity to work with him in many design projects. I am also grateful to all other members of the ICG, past, present, scientific and staff who are too numerous to name not only for the academic support but also for their friendship.

I would also like to thank my family. To my four parents, Aurora, Fernando, Elena and Raquel for providing me the guidance and support many could only dream of. To my siblings, Raquel, Manuel and Mary for providing me an example to follow. To my nieces and nephews Paulina, Alejandro, Emmanuel, Luis Andres and Renata for giving me inspiration and happiness. And my deepest thanks go to Doris without whom this dissertation would probably still be in the drawer and for being the place that now I call home.

Finally, I would like to thank you, my dear reader, for reading until the very end.

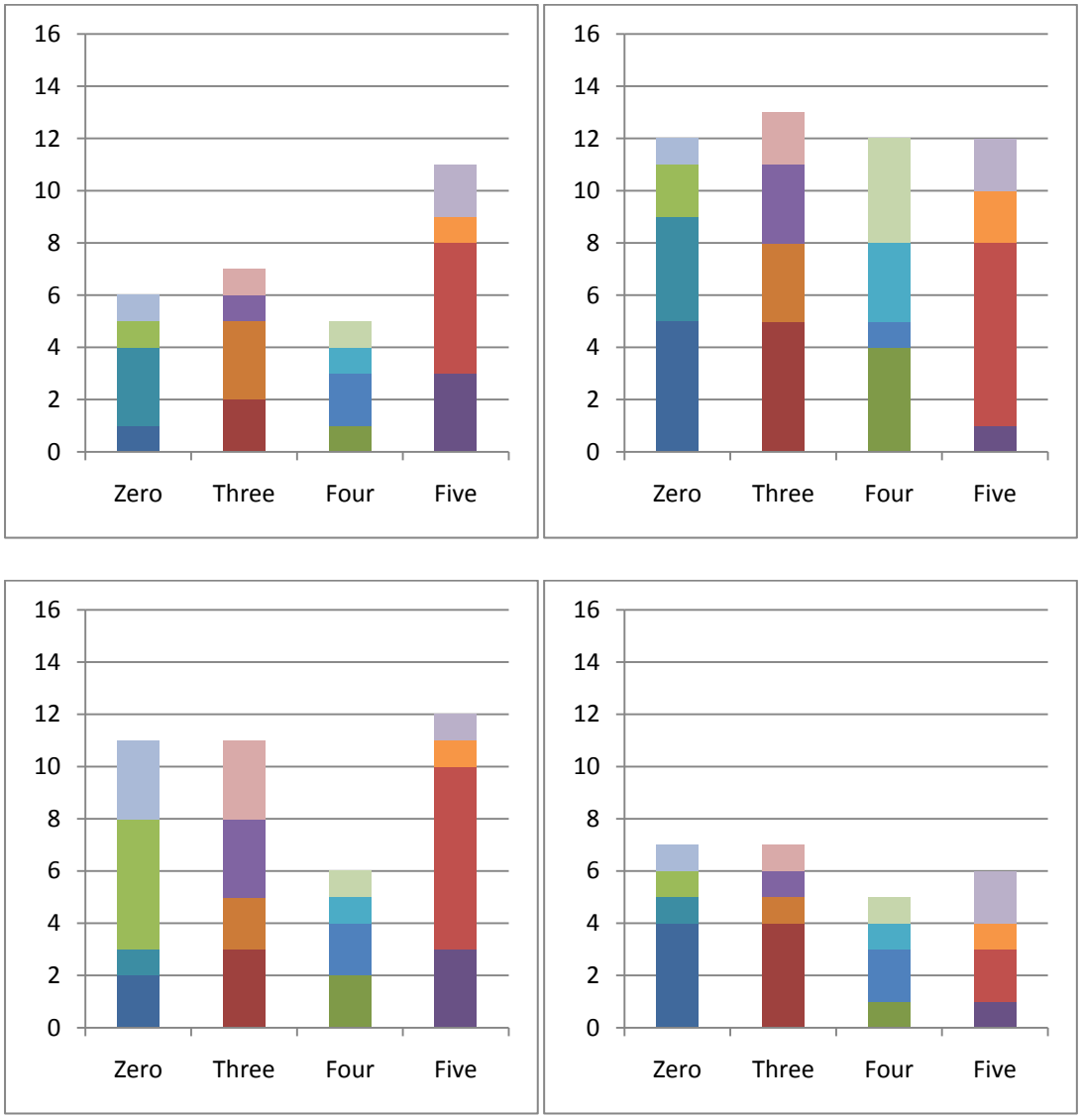
Chapter 8

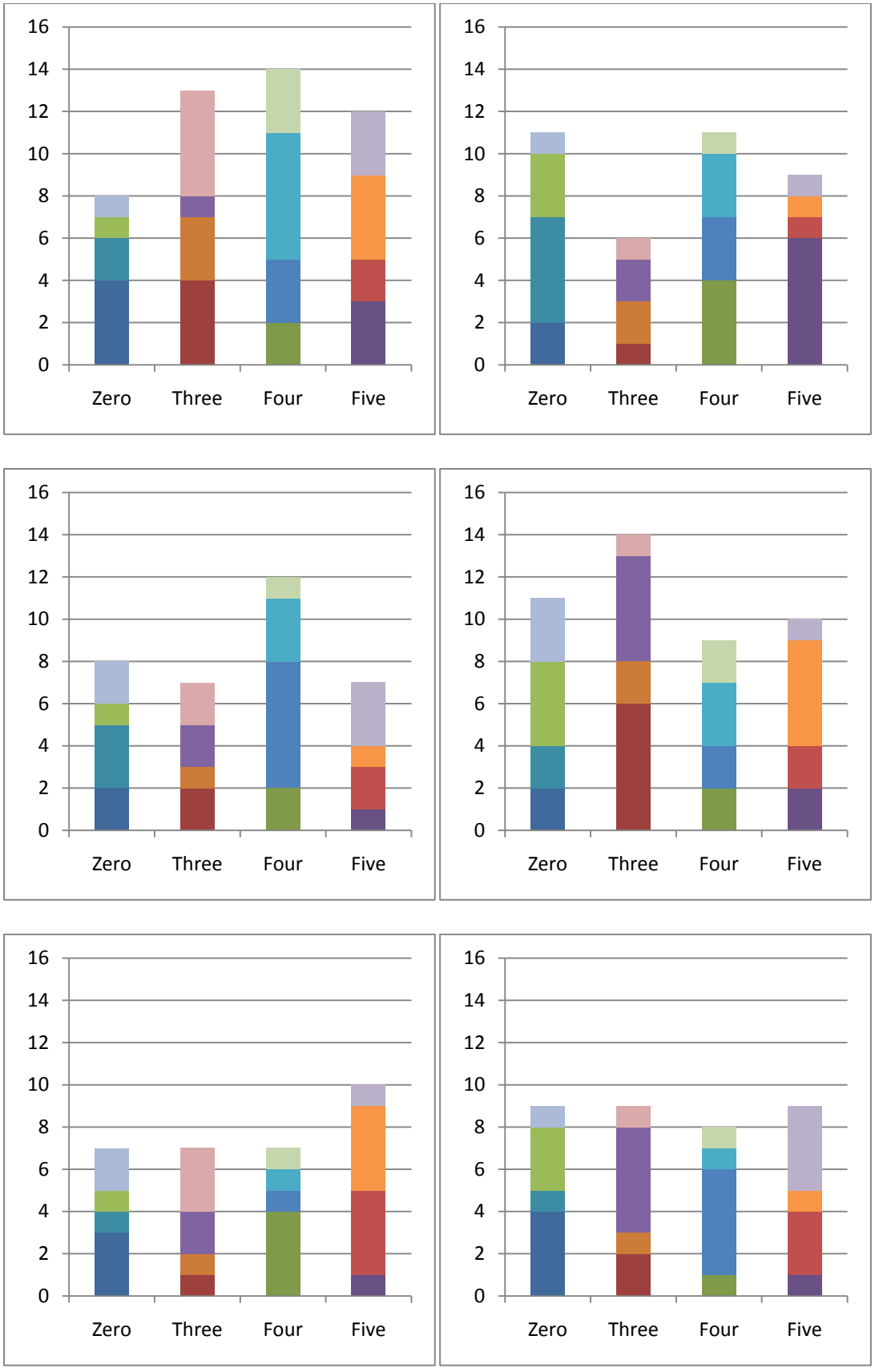
Appendix

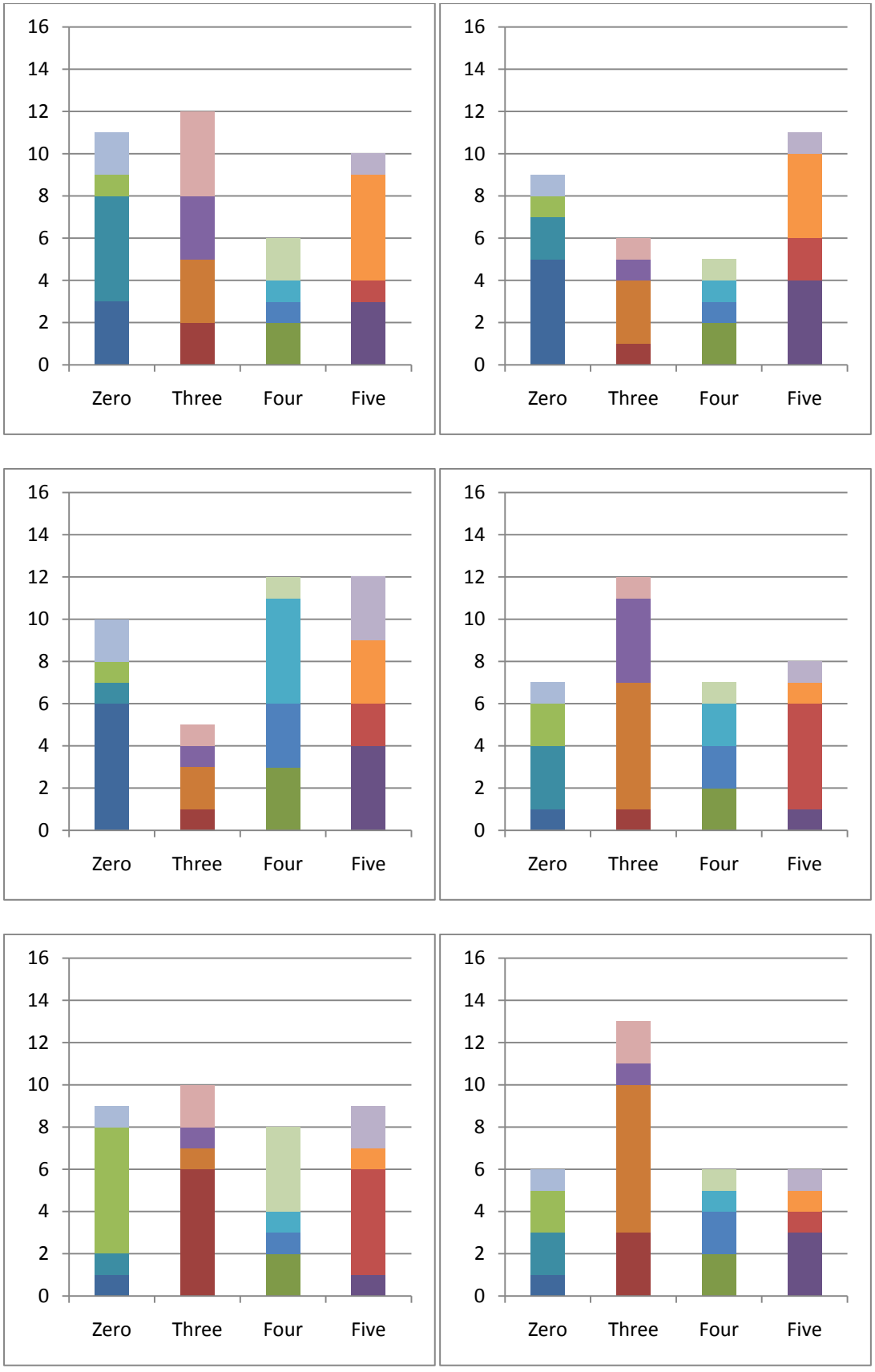
The following data is complementary to the rest of the dissertation.

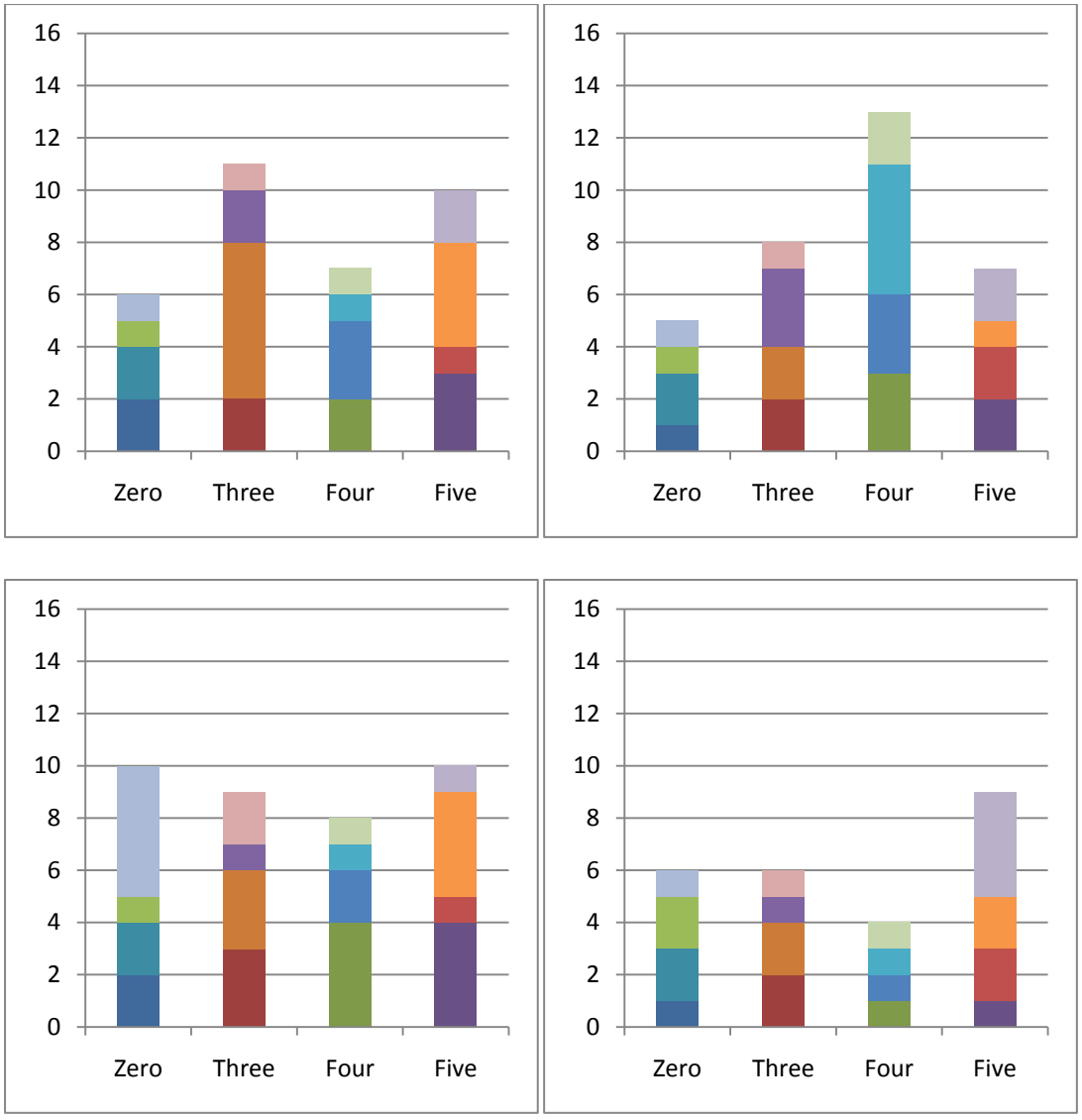
8.1 Responses for the awareness study

The following graphs represent the raw responses from the participants of the awareness user study. Every graph represents one video, thus there are twenty graphs. Every video is modulated with four different thresholds, thus every graph has four columns. The higher the score given to the video, the more "unnatural" it looked. Each response corresponds to a different participant.




16
14
12
10
8
6
4
2
0
Zero
Three
Four
Five


16
14
12
10
8
6
4
2
0
Zero
Three
Four
Five


16
14
12
10
8
6
4
2
0
Zero
Three
Four
Five
16
14
12
10
8
6
4
2
0
Zero
Three
Four
Five
16
14
12
10
8
6
4
2
0
Zero
Three
Four
Five
16
14
12
10
8
6
4
2
0
Zero
Three
Four
Five

Chapter 9

Bibliography

Abawi, Daniel F., Joachim Bienwald, and Ralf Dorner. 2004. Accuracy in Optical Tracking with Fiducial Markers: An Accuracy Function for ARToolKit. In *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, 260-261. Arlington, USA: IEEE Computer Society. http://portal.acm.org/citation.cfm?id=1032652.1033726.

Achanta, Radhakrishna, Francisco Estrada, Patricia Wils, and Sabine Süsstrunk. 2008. Salient Region Detection and Segmentation. In *Computer Vision Systems*,Antonios Gasteratos, Markus Vincze, and John K. Tsotsos, 5008:66-75. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg. doi:10.1007/978-3-540-79547-6. http://www.springerlink.com/content/9161v24g5442j61x.

Appel, Arthur, F. James Rohlf, and Arthur J. Stein. 1979. The haloed line effect for hidden line elimination. In *ACM International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 13:151-157. ACM Press. doi:10.1145/800249.807437.

Appel, Arthur. 1967. The notion of quantitative invisibility and the machine rendering of solids. *ACM Annual Conference*: 387-393.

Azuma, Ronald. 1997. A Survey of Augmented Reality. *Presence: Teleoperators and Virtual Environments* 6, no. 4: 355-385.

Bailey, Reynold, Ann McNamara, Nisha Sudarsanam, and Cindy Grimm. 2009. Subtle gaze direction. *ACM Transactions on Graphics (TOG)* 28, no. 4. http://portal.acm.org/citation.cfm?id=1559757.

Bane, Ryan, and Tobias Hoellerer. 2004. Interactive Tools for Virtual X-Ray Vision in Mobile Augmented Reality. In *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, 231-239. IEEE Computer Society. http://portal.acm.org/citation.cfm?id=1033720.

Barrows, Andrew K., and J. David Powell. 1999. Tunnel-in-the-sky cockpit display for complex remote sensing flight trajectories. In *International Airborne Remote Sensing Conference and Exhibition/21st Canadian Symposium on Remote Sensing*. Ottawa, Canada.

Beach, Richard, and Maureen C. Stone. 1983. Graphical style towards high quality illustrations. In *ACM International Conference on Computer Graphics and In-*

teractive Techniques (SIGGRAPH), 135. ACM. http://portal.acm.org/citation.cfm?id=964967.801141.

Bell, Blaine, Steven K. Feiner, and Tobias Hoellerer. 2001. View management for virtual and augmented reality. In *ACM Symposium on User Interface Software and Technology (UIST)*, 101-110. Orlando, USA. http://portal.acm.org/citation.cfm?id=502348.502363.

Benford, Steve, and Lennart Fahlen. 1993. A Spatial Model of Interaction in Large Virtual Environments. In *European Conference on Computer Supported Cooperative Work (ECSCW)*,G De Michelis, C Simone, and K Schmidt, 109-124. Kluwer Academic Publishers. http://www.ecscw.org/1993.htm.

Bergholm, Fredrik. 1987. Edge Focusing. *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-9, no. 6: 726-741. doi:10.1109/TPAMI.1987.4767980. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4767980.

Bertalmio, Marcelo, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. 2000. Image inpainting. *ACM International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*: 417-424. http://portal.acm.org/citation.cfm?id=344972.

Bichlmeier, Christoph, Felix Wimmer, Sandro M. Heining, and Nassir Navab. 2007. Contextual Anatomic Mimesis Hybrid In-Situ Visualization Method for Improving Multi-Sensory Depth Perception in Medical Augmented Reality. In *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, 129-138. Nara, Japan: IEEE Computer Society. http://portal.acm.org/citation.cfm?id=1514349.

Bichlmeier, Christoph, and Nassir Navab. 2006. Virtual Window for Improved Depth Perception in Medical AR. In *International Workshop on Augmented Reality Environments for Medical Imaging and Computer-aided Surgery (AMI-ARCS)*. Copenhagen, Denmark.

Bier, Eric A., Maureen C. Stone, Ken Pier, William Buxton, and Tony D. DeRose. 1993. Toolglass and magic lenses: the see-through interface. In *ACM International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 93pp:80. ACM. http://portal.acm.org/citation.cfm?id=166117.166126.

Biocca, Frank, Arthur Tang, Charles Owen, and Fan Xiao. 2006. Attention Funnel: Omnidirectional 3D Cursor for Mobile Augmented Reality Platforms. In *ACM Conference on Human Factors in Computing Systems (CHI)*, 06:1115-1122. ACM Press. doi:10.1145/1124772.1124939. http://portal.acm.org/citation.cfm?doid=1124772.1124939.

Bruckner, Stefan, and M. Eduard Groeller. 2007. Enhancing Depth-Perception with Flexible Volumetric Halos. *IEEE Transactions on Visualization and Computer*

Graphics (TVCG) 13, no. 6: 1344-1351. http://portal.acm.org/citation.cfm?id=1313166.

Canny, John. 1986. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8, no. 6: 679 - 698. http://portal.acm.org/citation.cfm?id=11275.

Caudell, Thomas P., and David W. Mizell. 1992. Augmented reality: an application of heads-up display technology to manual manufacturing processes. In *Hawaii International Conference on System Sciences*, 659-669. Hawaii: IEEE Computer Society. doi:10.1109/HICSS.1992.183317. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=183317.

Cerf, Moran, Jonathan Harel, Wolfgang Einhaueser, and Christof Koch. 2008. Predicting human gaze using low-level saliency combined with face detection. In *Advances in Neural Information Processing Systems*. Vancouver, British Columbia, Canada. http://books.nips.cc/papers/files/nips20/NIPS2007_1074.pdf.

Clark, James H. 1976. Hierarchical geometric models for visible-surface algorithms. *ACM International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)* 10, no. 2. http://portal.acm.org/citation.cfm?id=563274.563323.

Coelho, Enylton Machado, Blair MacIntyre, and Simon J. Julier. 2004. OSGAR: A Scene Graph with Uncertain Transformations. In *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, 6-15. Arlington, USA: IEEE Computer Society. http://portal.acm.org/citation.cfm?id=1032652.1033698.

Coelho, Enylton Machado. 2005. Spatially Adaptive Augmented Reality.

Collewijn, H., C. J. Erkelens, and R. M. Steinman. 1988. Binocular coordination of human horizontal saccadic eye movements. *The Journal of Physiology* 404: 157-82. http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=1190820&tool=pmcentrez&rendertype=abstract.

Danielsson, Per-Erik. 1980. Euclidean Distance Mapping. *Computer Graphics and Image Processing* 14, no. 3: 227-248.

DeCarlo, Doug, Adam Finkelstein, Szymon Rusinkiewicz, and Anthony Santella. 2003. Suggestive Contours for Conveying Shape. *ACM Transactions on Graphics (TOG)* 22, no. 3: 848-855.

Dey, Anind K. 2000. Providing Architectural Support for Building Context-Aware Applications.

Dey, Anind K. 2001. Understanding and Using Context. *Personal and Ubiquitous Computing* 5, no. 1: 4--7.

Diepstraten, Joachim, Daniel Weiskopf, and Thomas Ertl. 2003. Interactive cutaway illustration. *EUROGRAPHICS* 22, no. 3: 523-532.

Drozdek, Adam. 2004. *Data Structures and Algorithms in C++*. Course Technology. http://www.amazon.com/Data-Structures-Algorithms-Adam-Drozdek/dp/0534491820.

Engel, S., X. Zhang, and B. Wandell. 1997. Colour tuning in Human Visual Cortex Measured with Functional Magnetic Resonance Imaging. *Nature* 388, no. 6637: 68-71. doi:10.1038/40398. http://www.ncbi.nlm.nih.gov/pubmed/9214503.

Everitt, Cass. 2001. Interactive order-independent transparency. *White paper* 2, no. 6: 7. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.18.9286&rep=rep1&type=pdf.

Feiner, Steven K., Blair MacIntyre, and Dorée Duncan Seligmann. 1993. Knowledge-based augmented reality. *Communications of the ACM - Special Issue on Computer Augmented Environments: Back to the Real World* 36, no. 7: 53-62. doi:http://doi.acm.org/10.1145/159544.159587.

Feiner, Steven K., and Dorée Duncan Seligmann. 1992. Cutaways and ghosting: satisfying visibility constraints in dynamic 3D illustrations. *The Visual Computer* 8, no. 5-6: 292-302. doi:10.1007/BF01897116. http://www.springerlink.com/content/hun636803x7576tj.

Feiner, Steven K. 1985. Apex: An Experiment in the Automated Creation of Pictorial Explanations. *IEEE Computer Graphics and Applications (CG&A)* 5, no. 11: 29-37. http://portal.acm.org/citation.cfm?id=1299975.1300548.

Foley, James D., Andries van Dam, Steven K. Feiner, and John F. Hughes. 1995. *Computer Graphics: Principles and Practice in C (2nd Edition)*. Addison-Wesley Professional. http://www.amazon.com/Computer-Graphics-Principles-Practice-2nd/dp/0201848406.

Fuchs, Henry, Gregory D. Abram, and Eric D. Grant. 1983. Near real-time shaded display of rigid objects. *ACM International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)* 17, no. 3: 65-72. doi:10.1145/964967.801134. http://portal.acm.org/citation.cfm?doid=964967.801134.

Furmanski, Chris, Ronald Azuma, and Mike Daily. 2002. Augmented-Reality Visualizations Guided by Cognition: Perceptual Heuristics for Combining Visible and Obscured Information. In *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, 215. IEEE Computer Society. http://portal.acm.org/citation.cfm?id=850976.854989.

Goldstein, E. Bruce. 2006. *Sensation and Perception*. 7th. Wadsworth Publishing Company. http://www.amazon.com/Sensation-Perception-E-Bruce-Goldstein/dp/0534539645.

Goodwin, Todd, Ian Vollick, and Aaron Hertzmann. 2007. Isophote Distance: A Shading Approach to Artistic Stroke Thickness. In *International Symposium on Non-photorealistic Animation and Rendering (NPAR)*, 53-62.

Gouraud, Henri. 1971. Continuous Shading of Curved Surfaces. *IEEE Transactions on Computers* 20, no. 9: 623-629. http://portal.acm.org/citation.cfm?id=1310980.

Havemann, Sven. 2005. Generative mesh modeling. *PhD Thesis, Technische Universitaet Braunschweig, Germany*. http://generalized-documents.org/CGVold/DigitalLibrary/publications/TechnicalReports/bs/TR-tubs-cg-2003-01.pdf.

Heath, Michael, Sudeep Sarkar, Thomas Sanocki, and Kevin Bowyer. 1996. Comparison of edge detectors: a methodology and initial study. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)* 69, no. 1: 143-148. doi:10.1109/CVPR.1996.517066. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=517066.

Hering, Ewald. 1964. Outlines of a Theory of the Light Sense. *Harvard University Press*, no. 1905. http://psycnet.apa.org/psycinfo/1964-35010-000.

Hirani, Anil N., and Takashi Totsuka. 1996. Combining frequency and spatial domain information for fast interactive image noise removal. *ACM International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*: 269-276. http://portal.acm.org/citation.cfm?id=237170.237264.

Interrante, Victoria, Henry Fuchs, and Stephen M. Pizer. 1996. Illustrating transparent surfaces with curvature-directed strokes. In *IEEE Visualization*, 211-218. San Francisco, CA, United States: ACM. http://portal.acm.org/citation.cfm?id=245567.

Interrante, Victoria, Henry Fuchs, and Stephen M. Pizer. 1997. Conveying the 3D shape of smoothly curving transparent surfaces viatexture. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 3, no. 2: 98–117.

Itti, Laurent, Christof Koch, and Ernst Niebur. 1998. A Model of Saliency-Based Visual Attention for Rapid Scene Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20, no. 11: 1254-1259 . http://portal.acm.org/citation.cfm?id=297870.

Itti, Laurent, and Christof Koch. 1999. Comparison of feature combination strategies for saliency-based visual attention systems. In *SPIE*,Bernice E. Rogowitz

and Thrasyvoulos N. Pappas, 3644:473-482. San Jose, CA, USA: SPIE. http://link.aip.org/link/?PSI/3644/473/1.

Itti, Laurent, and Christof Koch. 2000. A saliency-based search mechanism for overt and covert shifts of visual attention. *Vision Research* 40, no. 10-12: 1489–1506. http://linkinghub.elsevier.com/retrieve/pii/S0042698999001637.

Itti, Laurent. 2007. Visual Salience. *Scholarpedia*.

Iverson, Lee A, and Steven W Zucker. 1995. Logical/Linear Operators for Image Curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 17, no. 10: 982-996. doi:10.110910.1109/34.464562. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=464562.

Johnson, Timothy E. 1963. Sketchpad III: a Computer Program for Drawing in Three Dimensions. *Spring Joint Computer Conference*: 347.

Judd, Tilke, Krista Ehinger, Fredo Durand, and Antonio Torralba. 2009. Learning to Predict Where Humans Look. In *IEEE International Conference on Computer Vision (ICCV)*. IEEE Computer Society.

Judd, Tilke, and Fredo Durand. 2007. Apparent Ridges for Line Drawing. *ACM Transactions on Graphics (TOG)* 26, no. 3: 19.

Juliano, Jeff, and Jeremy Sandmel. 2006. Framebuffer objects. *OpenGL Technical Paper*. http://oss.sgi.com/projects/ogl-sample/registry/EXT/framebuffer_object.txt.

Julier, Simon J., Yohan Baillot, Dennis Brown, and Marco Lanzagorta. 2002. Information Filtering for Mobile Augmented Reality. *IEEE Computer Graphics and Applications (CG&A)* 22, no. 5: 12-15. http://portal.acm.org/citation.cfm?id=616078.618925.

Kalkofen, Denis, Bernhard Reitinger, Risholm Petter, Alexander Bornik, Reinhard Beichel, Dieter Schmalstieg, and Samset Eigil. 2006. Integrated Medical Workflow for Augmented Reality Applications. In *International Workshop on Augmented Reality Environments for Medical Imaging and Computer-aided Surgery (AMI-ARCS)*. Copenhagen, Denmark.

Kalkofen, Denis, Erick Mendez, and Dieter Schmalstieg. 2007. Interactive Focus and Context Visualization for Augmented Reality. In *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, 191-200. Nara, Japan: IEEE Computer Society. http://portal.acm.org/citation.cfm?id=1514357.

Kalkofen, Denis, Erick Mendez, and Dieter Schmalstieg. 2009. Comprehensible Visualization for Augmented Reality. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 15, no. 2: 193-204. http://portal.acm.org/citation.cfm?id=1495798.1495994.

Kamada, Tomihisa, and Satoru Kawai. 1987. An Enhanced Treatment of Hidden Lines. *ACM Transactions on Graphics (TOG)* 6: 308-323.

Kim, Youngmin, and Amitabh Varshney. 2006. Saliency-guided Enhancement for Volume Visualization. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 12, no. 5: 925-932. http://portal.acm.org/citation.cfm?id=1187627.1187809.

Kim, Youngmin, and Amitabh Varshney. 2008. Persuading Visual Attention through Geometry. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 14, no. 4: 772-782. http://portal.acm.org/citation.cfm?id=1373109.1373249.

Koch, Christof, and Shimon Ullman. 1985. Shifts in selective visual attention: towards the underlying neural circuitry. *Human Neurobiology* 4, no. 4: 219-27. http://www.ncbi.nlm.nih.gov/pubmed/3836989.

Koenderink, Jan, and A. J. van Doorn. 1998. The Structure of Relief. *Advances in Imaging and Electron Physics* 103: 65-150.

Koenderink, Jan. 1990. *Solid Shape*. MIT Press.

Kokaram, A.C., R. D. Morris, W. J. Fitzgerald, and P. J. W. Rayner. 1995. Interpolation of missing data in image sequences. *ACM International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)* 4, no. 11: 1509 - 1519. http://ieeexplore.ieee.org/Xplore/login.jsp?url=http://ieeexplore.ieee.org/iel4/83/9915/00469932.pdf?arnumber=469932&authDecision=-203.

Kosara, Robert, Helwig Hauser, and Donna Gresh. 2003. An interaction view on information visualization. *EUROGRAPHICS*. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.10.1078&rep=rep1&type=pdf.

Kruger, Jens, Jens Schneider, and Rudiger Westermann. 2006. ClearView: An Interactive Context Preserving Hotspot Visualization Technique. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 12, no. 5: 941-948. http://portal.acm.org/citation.cfm?id=1187811.

Lee, Sungkil, Gerard Jounghyun Kim, and Seungmoon Choi. 2007. Real-time tracking of visually attended objects in interactive virtual environments. In *ACM Symposium on Virtual Reality Software and Technology (VRST)*, 38. ACM. http://portal.acm.org/citation.cfm?id=1315184.1315187.

Leykin, Alex, and Mihran Tuceryan. 2004. Determining Text Readability Over Textured Backgrounds in Augmented Reality Systems. In *Virtual Reality Continuum And Its Applications*, 436 - 439. ACM Press.

Li, Wilmot, Maneesh Agrawala, and David Salesin. 2004. Interactive image-based exploded view diagrams. In *Graphics Interface*, 62:203-212. Ontario, Canada: ACM Press. http://portal.acm.org/citation.cfm?id=1006083.

Livingston, Mark A., J. Edward Swan, Joseph L. Gabbard, Tobias Hoellerer, Deborah Hix, Simon J. Julier, Yohan Baillot, and Dennis Brown. 2003. Resolving Multiple Occluded Layers in Augmented Reality. In *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, 56. http://portal.acm.org/citation.cfm?id=946248.946796.

Longhurst, Peter, Kurt Debattista, and Alan Chalmers. 2006. A GPU based saliency map for high-fidelity selective rendering. In *International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*, 1:29. ACM. http://portal.acm.org/citation.cfm?id=1108595.

Looser, Julian, Mark Billinghurst, and Andy Cockburn. 2004. Through the looking glass: the use of lenses as an interface tool for Augmented Reality interfaces. In *International Conference on Computer Graphics and Interactive Techniques in Australiasia and South East Asia (GRAPHITE)*, 204-211. Singapore: ACM Press. http://portal.acm.org/citation.cfm?id=988870.

Lu, Boun Vinh, Tetsuya Kakuta, Rei Kawakami, Takeshi Oishi, and Katsushi Ikeuchi. 2010. Foreground and Shadow Occlusion Handling for Outdoor Augmented Reality. In *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*.

Ma, Kwan-Liu, and Victoria Interrante. 1997. Extracting Feature Lines from 3D Unstructured Grids. In *IEEE Visualization*, 285.

Malzebin, Pierre, Wayne Piekarski, Bruce Thomas, and Mark Billinghurst. 2002. Measuring ARToolKit accuracy in long distance tracking experiments. In *IEEE International Augmented Reality Toolkit Workshop*, 1-2. Darmstadt, Germany. http://arrow.unisa.edu.au:8080/vital/access/manager/Repository/unisa:36233.

Mark, William R., R. Steven Glanville, Kurt Akeley, and Mark J. Kilgard. 1993. Cg: a system for programming graphics hardware in a C-like language. *ACM International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*: 896-907. http://portal.acm.org/citation.cfm?id=882362.

Mendez, Erick, Denis Kalkofen, and Dieter Schmalstieg. 2006. Interactive context-driven visualization tools for augmented reality. In *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, 209-218. Santa Barbara, USA: IEEE Computer Society. http://portal.acm.org/citation.cfm?id=1514201.1514238.

Mendez, Erick, Dieter Schmalstieg, and Steven K. Feiner. 2010. Experiences on Attention Direction through Manipulation of Salient Features. In *IEEE Virtual Reality (VR) Workshop on Perceptual Illusions in Virtual Environments*, 4-9. Waltham, MA USA.

Mendez, Erick, Gerhard Schall, Sven Havemann, Sebastian Junghanns, Dieter W. Fellner, and Dieter Schmalstieg. 2008. Generating Semantic 3D Models of Under-

ground Infrastructure. *IEEE Computer Graphics and Applications (CG&A)* 28, no. 3: 48-57. http://portal.acm.org/citation.cfm?id=1373099.1373119.

Mendez, Erick, Steven K. Feiner, and Dieter Schmalstieg. 2010. Focus and Context in Mixed Reality by Modulating First Order Salient Features. In *International Symposium on Smart Graphics*, 232-243. Banff, Canada: Springer.

Mendez, Erick, and Dieter Schmalstieg. 2008. Context Sensitive Stylesheets for Scene Graphs. *International Journal of Virtual Reality (IJVR)* 7, no. 3: 77-84. http://www.ijvr.org/issues/issue3-2008/13.pdf.

Mendez, Erick, and Dieter Schmalstieg. 2009. Importance masks for revealing occluded objects in augmented reality. In *ACM Symposium on Virtual Reality Software and Technology (VRST)*, 247-248. Kyoto, Japan. http://portal.acm.org/citation.cfm?id=1643928.1643988.

Microsoft. 2010. Xbox.com | Kinect. http://www.xbox.com/en-US/kinect.

Milgram, Paul, and Fumio Kishino. 1994. A Taxonomy of Mixed Reality Visual Displays. *IEICE Transactions on Information Systems* E77-D, no. 12. http://vered.rose.utoronto.ca/people/paul_dir/IEICE94/ieice.html.

Nalwa, Vishvjit S., and Thomas O. Binford. 1986. On Detecting Edges. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8, no. 6: 699-714.

Niebur, Ernst. 2010. Saliency map. *Scholarpedia*. http://www.scholarpedia.org/article/Saliency_map.

Nitzberg, Mark, David Mumford, and Takahiro Shiota. 1993. *Filtering, Segmentation, and Depth*. Springer. http://portal.acm.org/citation.cfm?id=563003.

Ouerhani, Nabil, Roman Von Wartburg, Heinz Hugli, and Rene Muri. 2004. Empirical validation of the saliency-based model of visual attention. *Electronic Letters on Computer Vision and Image Analysis* 3, no. 1: 13–24. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.96.7278&rep=rep1&type=pdf.

Parkhurst, Derrick, Klinton Law, and Ernst Niebur. 2002. Modeling the role of salience in the allocation of overt visual attention. *Vision Research* 42, no. 1: 107-123. http://www.ncbi.nlm.nih.gov/pubmed/11804636.

Perlin, Ken. 1985. An image synthesizer. *ACM International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)* 19, no. 3: 287 - 296. http://portal.acm.org/citation.cfm?id=325247.

Phong, Bui Tuong. 1975. Illumination for Computer Generated Pictures. *Communications of the ACM* 18, no. 6: 311-317.

Pock, Thomas, Markus Grabner, and Horst Bischof. 2007. Real-time computation of variational methods on graphics hardware. In *Computer Vision Winter Workshop*.

Reitmayr, Gerhard, and Dieter Schmalstieg. 2001. Opentracker-an open software architecture for reconfigurable tracking based on XML. In *IEEE Virtual Reality (VR)*, 285–286. http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Opentracker+an+open+software+architecture+for+reconfigurable+tracking+based+on+XML#0.

Reitmayr, Gerhard, and Dieter Schmalstieg. 2005. Flexible Parametrization of Scene Graphs. In *IEEE Virtual Reality (VR)*, 51-58. Bonn, Germany. http://portal.acm.org/citation.cfm?id=1079827.

Rensink, Ronald A, J. Kevin O'Regan, and James J. Clark. 2000. On the Failure to Detect Changes in Scenes Across Brief Interruptions. *Visual Cognition* 7, no. 1-3: 127-145. doi:10.1080/135062800394720. http://www.informaworld.com/openurl?genre=article&doi=10.1080/135062800394720&magic=crossref.

Robertson, Cindy M., Blair MacIntyre, and Bruce N. Walker. 2007. An evaluation of graphical context as a means for ameliorating the effects of registration error. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 15, no. 2: 179-192. http://portal.acm.org/citation.cfm?id=1514339.1514344.

Rong, Guodong, and Tiow-Seng Tan. 2006. Jump Flooding in GPU with Applications to Voronoi Diagram and Distance Transform. In *ACM Symposium on Interactive 3D Graphics and Games*, 109-116.

Ropinski, Timo, and Klaus Hinrichs. 2004. Real-Time Rendering of 3D Magic Lenses having arbitrary convex Shapes. *International Winter School of Computer Graphics (WSCG)*: 379-386.

Rosenholtz, Ruth. 1999. A simple saliency model predicts a number of motion popout phenomena. *Vision Research* 39: 3157–3163. http://wexler.free.fr/library/files/rosenholtz (1999) a simple saliency model predicts a number of motion popout phenomena.pdf.

Rost, Randi J. 2004. *OpenGL Shading Language*. 1st editio. Pearson Education, Inc.

Rothwell, C., J. Mundy, W. Hoffman, and V. D. Nguyen. 1995. Driving Vision by Topology. In *IEEE International Symposium on Computer Vision*, 395-400. ftp://ftp-sop.inria.fr/robotvis/pub/html/crothwel/papers/cvs95.ps.gz.

Rusinkiewicz, Szymon, Forrester Cole, Doug DeCarlo, and Adam Finkelstein. 2008. Line Drawings from 3D Models. In *ACM International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*. ACM Press.

SIM. 2010. Coin3D. http://www.coin3d.org/.

Sangwie, Stephen J. 1998. *The Colour Image Processing Handbook (Optoelectronics, Imaging and Sensing)*. First. Chapman and Hall. http://www.amazon.com/Processing-Handbook-Optoelectronics-Imaging-Sensing/dp/0412806207.

Santella, Anthony, and Doug DeCarlo. 2004. Visual interest and NPR: an evaluation and manifesto. In *International Symposium on Non-photorealistic Animation and Rendering (NPAR)*, 150. ACM. http://portal.acm.org/citation.cfm?id=987657.987669.

Schmalstieg, Dieter, Anton Fuhrmann, Gerd Hesina, Zsolt Szalavári, L. Miguel Encarnacao, Michael Gervautz, and Werner Purgathofer. 2002. The studierstube augmented reality project. *Presence: Teleoperators and Virtual Environments* 11, no. 1: 33-54. http://portal.acm.org/citation.cfm?id=641636.

Schwerdtfeger, Bjorn, and Gudrun Klinker. 2008. Supporting order picking with Augmented Reality. In *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, 91-94. Cambridge, UK: IEEE Computer Society. http://portal.acm.org/citation.cfm?id=1605352.

Shreiner, Dave, Mason Woo, Jackie Neider, and Tom Davis. 2005. *OpenGL(R) Programming Guide: The Official Guide to Learning OpenGL(R), Version 2 (5th Edition)*. Addison-Wesley Professional. http://www.amazon.com/OpenGL-Programming-Guide-Official-Learning/dp/0321335732.

Siltanen, Sanni, and Charles Woodward. 2006. Augmented interiors with digital camera images. In *Australasian User Interface Conference*, 33-36. Hobart, Australia. http://portal.acm.org/citation.cfm?id=1151761.

Spillman, Lothar, and John S. Werner. 1990. *Visual Perception: The Neurophysiological Foundations*. Academic Press Inc. http://www.amazon.co.uk/Visual-Perception-Neurophysiological-Lothar-Spillman/dp/0126576769.

Strauss, Paul S., and Rikk Carey. 1992. An object-oriented 3D graphics toolkit. In *ACM International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 341-349. http://portal.acm.org/citation.cfm?id=133994.134089.

Su, Sara L., Fredo Durand, and Maneesh Agrawala. 2005. De-emphasis of distracting image regions using texture power maps. *Massachusetts Institute of Technology Computer Science and Artificial Intelligence Laboratory*: 119-124. http://18.51.3.32/handle/1721.1/30537.

Sutherland, Ivan. 1968. A Head-Mounted Three Dimensional Display. In *Fall Joint Computer Conference*, 757-764. Spartan Books.

Treisman, Anne M., and Garry Gelade. 1980. A feature-integration theory of attention. *Cognitive Psychology* 12, no. 1: 97-136. http://www.ncbi.nlm.nih.gov/pubmed/7351125.

Viega, John, Matthew J. Conway, George Williams, and Randy Pausch. 1996. 3D magic lenses. In *ACM Symposium on User Interface Software and Technology (UIST)*, 51-58. ACM. http://portal.acm.org/citation.cfm?id=237098.

Viola, Ivan, Armin Kanitsar, and M. Eduard Groeller. 2005. Importance-Driven Feature Enhancement in Volume Visualization. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 11, no. 4: 408-418. http://www.computer.org/portal/web/csdl/doi/10.1109/TVCG.2005.62.

Wang, Lujin, Ye Zhao, Klaus Mueller, and Arie Kaufman. 2005. The Magic Volume Lens: An Interactive Focus+Context Technique for Volume Rendering. In *IEEE Visualization*, 47. http://www.computer.org/portal/web/csdl/doi/10.1109/VIS.2005.100.

Ward, Lawrence M. 2008. Attention. *Scholarpedia*.

Wernecke, Josie. 1994. *The Inventor Mentor: Programming Object-Oriented 3D Graphics with Open Inventor*. Addison-Wesley Professional.

Williams, Lance. 1983. Pyramidal parametrics. *ACM International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)* 17, no. 3: 1-11. http://portal.acm.org/citation.cfm?id=801126.